Preprocessing for digital video using mathematical morphology

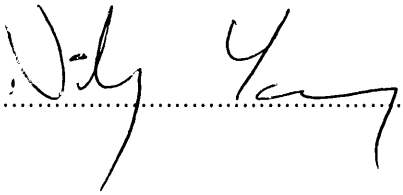
Young, Nicky

Award date:
2003

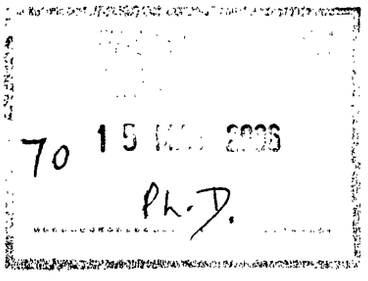
Awarding institution:
University of Bath

Preprocessing for Digital Video using Mathematical Morphology

Submitted by
Nicky Young
for the degree of
Doctor of Philosophy
of the
University of Bath 2003



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346


Ph.D.

Abstract

This thesis describes the work covered for the PhD research project, *"Preprocessing for digital video using Mathematical Morphology"*. The thesis details the history and principles behind digital video. Current methods of noise reduction and preprocessing are reviewed with the main focus of the research being the application of mathematical morphology to digital video processing, specifically for the use of noise reduction and preprocessing for image simplification. In addition, this thesis describes the principles behind mathematical morphology and details how it can be applied to digital video processing.

A detailed description of the development of a psychovisually lossless preprocessing system for digital image simplification using mathematical morphology is given. This details the techniques and methods used to develop the system and a full evaluation, both objective and subjective is given. In addition a noise reduction system is implemented for spatial processing of images and spatio-temporal processing of digital video sequences. Evaluations are made using an objective measure with two image CODEC's, JPEG and JPEG 2000. The performance of the system for video compression is also evaluated using an objective measure and four video CODEC's, MJPEG, MPEG-II, Bath Wavelet Video and H263. Application to real-world data, where little or no prior information is known is also detailed for both spatial and spatio-temporal systems. Finally conclusions are drawn and possible areas of future development are identified.

Contents

1 INTRODUCTION 1

2 INTRODUCTION TO VIDEO 3

2.1 HISTORY OF IMAGING 3
2.2 ANALOGUE VIDEO 5
2.2.1 Analogue Colour Representation 6
2.3 DIGITAL VIDEO 8
2.3.1 Bandwidth of Analogue and Digital Video 10
2.3.2 Sources of Redundancy in Video 10
2.3.3 Still Image Compression 13
2.3.4 Digital Colour Representation 14
2.3.5 Video Compression 16
2.3.6 Moving Pictures Experts Group Video CODEC 18
2.4 CONCLUSIONS 19

3 THE HUMAN VISUAL SYSTEM 21

3.1 VISUAL PERCEPTION 21
3.2 VISUAL EVALUATION 22
3.2.1 Objective Evaluation 22
3.2.2 Subjective Evaluation 26
3.2.2.1 Double Stimulus, Impairment Scale 26
3.2.2.2 Double Stimulus, Continuous Quality Scale 27
3.2.2.3 Other Methods 28
3.2.2.4 Analysing the Results 28
3.3 CONCLUSION 29

4 MATHEMATICAL MORPHOLOGY 30

4.1 BINARY MORPHOLOGY 30
4.1.1 Dilation 30
4.1.2 Erosion 32
4.1.3 Opening and Closing 34
4.2 GREYSCALE MORPHOLOGY 36
4.2.1 Greyscale Dilation 36
4.2.2 Greyscale Erosion 37
4.3 TWO DIMENSIONAL MORPHOLOGY 38
4.3.1 2D Dilation 38
4.3.2 2D Erosion 38

4.3.3 Application to Image Filtering *40*
4.4 THREE DIMENSIONAL MORPHOLOGY 40
4.4.1 3D Dilation *40*
4.4.2 3D Erosion *44*
4.4.3 Application to Video Preprocessing *44*
4.5 GRANULOMETRIES 48
4.6 ATTRIBUTE MORPHOLOGY 52
4.6.1 Area Morphology *52*
4.6.2 Filter Structures *53*
4.6.3 Generalisation to Other Attributes *54*
4.6.4 Application to Video *30*
4.7 CONCLUSION 56

5 NOISE IN DIGITAL IMAGES 57

5.1 SOURCES OF NOISE IN DIGITAL VIDEO 57
5.1.1 Gaussian Noise Model *58*
5.1.2 Impulse Noise *58*
5.1.3 Exponential Noise *59*
5.1.4 Uniform Noise *59*
5.2 NOISE ESTIMATION 61
5.3 NOISE REDUCTION 62
5.3.1 Linear Filtering *62*
5.3.1.1 Homomorphic Filtering 65
5.3.1.2 Gaussian Smoothing 66
5.3.2 Non-linear Filtering *69*
5.3.2.1 Median Filtering 69
5.3.2.2 Outlier Filter 71
5.3.2.3 Image Restoration 72
5.3.3 Review of Current Image Noise Reduction Techniques *73*
5.3.4 Mathematical Morphology for Noise Reduction *77*
5.3.4.1 Morphological Image Cleaning 78
5.4 PREPROCESSING 79
5.5 CONCLUSION 80

6 IMAGE PREPROCESSING USING ATTRIBUTE MORPHOLOGY 82

6.1 PIXEL QUEUE ALGORITHM 85
6.1.1 Morphological Reconstruction *85*
6.1.1.1 Extending to Greyscale 87
6.1.1.2 Extending to 2D 89
6.1.1.3 Sequential Reconstruction 91
6.1.1.4 FIFO Reconstruction 91
6.1.1.5 Hybrid Reconstruction 93

6.1.2 Region Growing *95*
6.1.2.1 Hierarchical Queues 95
6.1.2.2 Priority Queues 97
6.1.3 Performance of the Pixel Queue Method *98*
6.1.4 Max-tree *101*
6.1.5 Wilkinson's Method *106*
6.2 ATTRIBUTE MORPHOLOGY 107
6.2.1 Pixel Queue Algorithm for Attributes *109*
6.2.2 Attributes *110*
6.2.2.1 Area 110
6.2.2.2 Contrast 110
6.2.2.3 Volume 110
6.2.2.4 Power 111
6.3 FILTER STRUCTURES 112
6.3.1 Efficient ASF Implementation *112*
6.4 CONCLUSION 115

7 PREPROCESSOR DEVELOPMENT 116

7.1 IMAGE NOISE REDUCTION 116
7.1.1 Gaussian Filtering *116*
7.1.1.1 Compression Results 124
7.1.2 Morphological Filtering *130*
7.1.2.1 Compression Results 153
7.1.3 Application to Unseen Data *155*
7.1.3.1 Estimating Noise Variance 158
7.1.3.2 Applying Filters to Unseen Data 163
7.1.3.3 Compression Results 171
7.2 PSYCHOVISUALLY LOSSLESS SIMPLIFICATION 175
7.2.1 Simplification *175*
7.2.2 Psychovisual Determination of Visually Lossless Attribute Values *180*
7.2.3 CODEC Output Evaluation *190*
7.2.4 Application to Unseen Data *199*
7.2.4.1 CODEC Performance on Unseen Data 207
7.3 CONCLUSION 214

8 INTERMEDIATE AREA-MORPHOLOGY 215

8.1.1 Implementing Intermediate Area-Open-Close *217*
8.1.2 Intermediate Morphological Gradient *221*
8.1.3 Intermediate Distance *224*
8.1.4 Full Intermediate Area Morphology *227*
8.2 CONCLUSION 229

9 3D PREPROCESSOR DEVELOPMENT FOR NOISE REDUCTION 230

9.1 SPATIAL PROCESSING OF VIDEO 230
9.1.1 Gaussian Filtering *231*
9.1.2 K-Nearest Neighbour Filtering *237*
9.1.3 Morphological Filtering *243*
9.1.4 Compression Results *248*
9.2 SPATIO-TEMPORAL MORPHOLOGICAL FILTERING 250
9.2.1 Non-Overlapping Window Processing *250*
9.2.1.1 Compression Results 252
9.2.2 Overlapping Window Processing *255*
9.2.2.1 Compression Results 257
9.3 HYBRID SPATIAL AND SPATIO-TEMPORAL MORPHOLOGICAL FILTERING 261
9.3.1 Combined Spatial and Spatio-Temporal Method *261*
9.3.1.1 Compression Results 266
9.3.2 Simultaneous Spatial and Spatio-Temporal Method *271*
9.3.2.1 Compression Results 273
9.3.2.2 Visual Evaluation of the Simultaneous Morphological Method 279
9.4 PSYCHOVISUAL EVALUATION 289
9.5 APPLICATION TO UNSEEN DATA 290
9.5.1 Estimating the Noise Variance *292*
9.5.2 Filtering Unseen Data *292*
9.5.3 Compression Results *300*
9.6 CONCLUSION 301

10 CONCLUSIONS 304

11 ACKNOWLEDGEMENTS 308

12 APPENDIX A - SET THEORY 309

12.1 SETS 309
12.1.1 Venn Diagrams *309*
12.1.2 Intersection *311*
12.1.3 Union *312*
12.1.4 Compliment *312*
12.1.5 Set Difference *313*
12.1.6 Equality *314*
12.1.7 Translation *314*
12.1.8 Reflection *314*
12.2 PREDEFINED SETS 315
12.3 BASIC LAWS 315
12.4 OTHER USEFUL OPERATORS 316

13 APPENDIX B – FULL RESULTS 317

13.1 IMAGE NOISE REDUCTION 317

13.1.1 Gaussian filtering results *317*
13.1.2 Morphology filtering results *325*
13.1.3 Compression Results *351*
13.1.4 Application to unknown data *355*
13.2 IMAGE SIMPLIFICATION 357

14 AUTHOR'S PUBLICATIONS **369**

15 REFERENCES **370**

List of Figures

Figure 2.1: The illusion of motion. 3
Figure 2.2: Progressive and interlaced scanning 6
Figure 2.3: The relationship between the three primary colour representations used in the analogue video system. 7
Figure 2.4: Analogue to digital signal conversion. 9
Figure 2.5: Spatial redundancy in images. The two blocks marked in the sky and on the sand are almost a constant colour and hence contain little information. 11
Figure 2.6: The temporal redundancy of video 12
Figure 2.7: Arrangement of a typical CODEC. 15
Figure 2.8: Basic sub-sampling. 16
Figure 2.9: RGB to YCrCb 4:2:2 colour conversion process. 17
Figure 2.10: Example of how frames are grouped together in MPEG-II 18
Figure 2.11: Compression artefacts of frame 25 from the 352 x 288, YUV 4:2:0 (12bpp) Stefan sequence. 20
Figure 3.1: Demonstrations of visual illusions. 21
Figure 3.2: Example of the Webber ratio. 22
Figure 3.3: The use of objective measurements 24
Figure 3.4: Example of the problems with objective measures, which say that JPEG (c) is better than the noisy image(b). However, psychovisually, the noisy image (b) is often preferred over the JPEG (c). 25
Figure 3.5: Stages of visual testing using the DSIS method. 27
Figure 3.6: The scoring scale used in DSCQS 28
Figure 4.1: An example of the Dilation operation. 32
Figure 4.2: Example of how erosion works. 34
Figure 4.3: The opening operator. 35
Figure 4.4: The closing operator 35
Figure 4.5: The Open-Close and Close-Open operators. 36
Figure 4.6: An example of greyscale dilation. 37
Figure 4.7: A variation on the standard dilation using a non-flat structuring element 37
Figure 4.8: 2D extension of the structuring element 38
Figure 4.9: An example of 2D dilation. 39
Figure 4.10: Basic morphological operations, dilation and erosion applied to the 720 x 576, 8bit greyscale Barbara 2 image. 41
Figure 4.11: Basic morphological operations, closing and opening applied to the 720 x 576, 8bit greyscale Barbara 2 image. 42
Figure 4.12: Basic morphological operations, close-open and open-close applied to the 720 x 576, 8bit greyscale Barbara 2 image. 43

Figure 4.13: Extension of the 4nn (left) and 8nn (right) to 3D giving the 6nn (or 4_{nn}^{3D}) and 26nn (or 8_{nn}^{3D}) respectively. 44
Figure 4.14: Basic principles of 3D morphology 45
Figure 4.15: Example of the effects of 3D dilation on low motion video 46
Figure 4.16: Example of the effects of 3D dilation on fast motion video. 47
Figure 4.17: Increasing size structuring elements with the origin shaded. 48
Figure 4.18: The sieve structure. 50
Figure 4.19: Sieve example of size 2. 51
Figure 4.20: Increasing square structuring element. 51
Figure 4.21: Example of A_λ for 4nn and λ=2. Shaded cell is the origin. 52
Figure 4.22: Example of A_λ for 8nn and λ=2. Shaded cell is the origin. 53
Figure 4.23: 2D sieving example using an artifitial image. 55
Figure 5.1: Gaussian and Impulse noise model PDF (left) and resultant image (right). 60
Figure 5.2: Exponential and Uniform noise model PDF (left) and resultant image (right). 61
Figure 5.3: Image filtering using a mask. 64
Figure 5.4: Impulse noise filtering using the mean filter and the 720 x 576, 8bit greyscale Barbara image. 65
Figure 5.5: Gaussian noise filtering using the mean filter and the 720 x 576, 8bit greyscale Barbara image. 66
Figure 5.6: Homomorphic filtering example using the 720 x 576, 8bit greyscale Barbara image. 67
Figure 5.7: A Gaussian filter mask. 67
Figure 5.8: Impulse noise filtering using the Gaussian filter and the 720 x 576, 8bit greyscale Barbara image. 68
Figure 5.9: Gaussian noise filtering using the Gaussian filter and the 720 x 576, 8bit greyscale Barbara image. 69
Figure 5.10: Impulse noise filtering using the median filter and the 720 x 576, 8bit greyscale Barbara image. 71
Figure 5.11: Gaussian noise filtering using the median filter and the 720 x 576, 8bit greyscale Barbara image. 72
Figure 5.12: Example of using the K-NN filter. 76
Figure 6.1: Example to show local and regional maximas. 83
Figure 6.2: Example of how Area-Opening works. 83
Figure 6.3: A greyscale Area-Opening on the 720 x 576, 8bit greyscale Barbara 2 image. 84
Figure 6.4: Geodesic distance between points p and q within set X. 86
Figure 6.5: Maxima extraction. 88
Figure 6.6: Dome extraction. 89
Figure 6.7: Maxima extraction with fluctuations. 89
Figure 6.8: How to decompose the structuring element for efficient implementation of reconstruction. 92
Figure 6.9: Average execution time against number of pixels in the image for 50 images of each size. 94
Figure 6.10: Extracted regions using reconstruction on the 720 x 576, 8bit greyscale Barbara 2 image. 82
Figure 6.11: A priority queue. 97

Figure 6.12: Example of how a new node can be added to the queue. 99
Figure 6.13: Removal of a node from the queue. 100
Figure 6.14: Timing results for the pixel queue method using 50 8bit greyscale images with a size of 720 x 576 pixels. Area size was measured at intervals of 100 101
Figure 6.15: A binary Max-tree. 101
Figure 6.16: A greyscale Max-tree. 103
Figure 6.17: Timing results for the Max-tree method 105
Figure 6.18: Timing results for Wilkinson's method using 50 8bit greyscale, 720 x 576 images. Measured at intervals of an area size of 100 starting at 1 and ending at 10001. 109
Figure 6.19: Comparison of different attributes. 111
Figure 6.20: Timings (in Seconds) for ASF implementations using 4nn connectivity, 50 8bit greyscale images with a size of 720 x 576. Measurements have been made at area size intervals of 2, starting at 1 and ending at 51. 113
Figure 6.21: Timings (in Seconds) for ASF implementations using 8nn connectivity, 50 8bit greyscale images with a size of 720 x 576. Measurements have been made at area size intervals of 2, starting at 1 and ending at 51. 113
Figure 7.1: Gaussian results showing PSNR against sigma and mask size with 14dB of noise. 117
Figure 7.2: Gaussian results showing PSNR against sigma and mask size with 21dB of noise. 118
Figure 7.3: Gaussian results showing PSNR against sigma and mask size with 28dB of noise. 119
Figure 7.4: Gaussian results showing PSNR against sigma and mask size with 35dB of noise. 120
Figure 7.5: Filtering the Barbara image (8bit greyscale at 720 x 576 pixels) using the optimum Gaussian filter. 122
Figure 7.6: Filtering the Barbara image (8bit greyscale at 720 x 576 pixels) using the optimum Gaussian filter. 123
Figure 7.7: Barbara image (720 x 576, 8bit greyscale) compressed using the JPEG CODEC with a compression ratio of 20:1 (0.4bpp) and 14dB noise. 126
Figure 7.8: Barbara image (720 x 576, 8bit greyscale) compressed using the JPEG 2000 CODEC with a compression ratio of 20:1 (0.4bpp) and 14dB of noise 127
Figure 7.9: Barbara image (720 x 576, 8bit greyscale) filtered and compressed using the JPEG CODEC with a compression ratio of 20:1 (0.4bpp) and 35dB of noise. 128
Figure 7.10: Barbara image (720 x 576, 8bit greyscale) filtered and compressed using the JPEG 2000 CODEC with a compression ratio of 20:1 (0.4bpp) and 35dB of noise 129
Figure 7.11: Filter performance with respect to an increasing area size using an image corrupted with 14dB of noise 132
Figure 7.12: Filter performance with respect to an increasing contrast attribute using an image corrupted with 14dB of noise 133
Figure 7.13: Filter performance with respect to an increasing volume attribute using an image corrupted with 14dB of noise 134
Figure 7.14: Filter performance with respect to an increasing power attribute using an image corrupted with 14dB of noise 135
Figure 7.15: Filter performance with respect to an increasing area attribute using an image corrupted with 21dB of noise 136

Figure 7.16: Filter performance with respect to an increasing contrast attribute using an image corrupted with 21dB of noise.....137
Figure 7.17: Filter performance with respect to an increasing volume attribute using an image corrupted with 21dB of noise.....138
Figure 7.18: Filter performance with respect to an increasing power attribute using an image corrupted with 21dB of noise.....139
Figure 7.19: Filter performance with respect to an increasing area attribute using an image corrupted with 28dB of noise.....140
Figure 7.20: Filter performance with respect to an increasing contrast attribute using an image corrupted with 28dB of noise.....141
Figure 7.21: Filter performance with respect to an increasing volume attribute using an image corrupted with 28dB of noise.....142
Figure 7.22: Filter performance with respect to an increasing power attribute using an image corrupted with 28dB of noise.....143
Figure 7.23: Filter performance with respect to an increasing area attribute using an image corrupted with 35dB of noise.....144
Figure 7.24: Filter performance with respect to an increasing contrast attribute using an image corrupted with 35dB of noise.....145
Figure 7.25: Filter performance with respect to an increasing volume attribute using an image corrupted with 35dB of noise.....146
Figure 7.26: Filter performance with respect to an increasing power attribute using an image corrupted with 35dB of noise.....147
Figure 7.27: Corrupted input images of Barbara (720 x 576. 8bit greyscale) with AWGN of 14dB and 35dB that are used as the input to the morphological filter. 150
Figure 7.28: Morphological filter output using the Barbara image (720 x 576. 8bit greyscale) with 14dB of noise added.....151
Figure 7.29: Morphological filter output using the Barbara image (720 x 576. 8bit greyscale) with 35dB of noise added.....152
Figure 7.30: Gaussian sigma (σ) value against the noise variance. Points show the original measure data and the solid line shows a quadratic fit to the data. 158
Figure 7.31: Gaussian mask value against the noise variance. Points show the original measure data and the solid line shows a quadratic fit.....159
Figure 7.32: Area size value against the noise variance for 4nn connectivity and the AF filter structure. Points show the original measure data and the solid line shows a quadratic fit.....159
Figure 7.33: Frequency of the best percentage of results and block sizes to use.....160
Figure 7.34: Frequency of percentage of blocks to use in conjunction with a 2 x 2 block size.....162
Figure 7.35: Frequency of block sizes to use in conjunction with using 2% of the blocks. 162
Figure 7.36: Corrupted and filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) using 14dB of noise. 166
Figure 7.37: Corrupted and filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) using 35dB of noise. 167

Figure 7.38: Corrupted input images of Blackboard (720 x 576, 8bit greyscale) with AWGN of 14dB and 35dB that are used as the input to the morphological filter. 168
Figure 7.39: Filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) corrupted with 14dB of noise and using morphological filtering. 169
Figure 7.40: Filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) corrupted with 35dB of noise and using morphological filtering. 170
Figure 7.41: JPEG rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the AF filter structure. 176
Figure 7.42: JPEG rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the ASF filter structure. 177
Figure 7.43: JPEG 2000 rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the AF filter structure. 178
Figure 7.44: JPEG 2000 rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the ASF filter structure. 179
Figure 7.45: Psychovisual test results for 4nn connectivity using the AF filtering structure and the area attribute. The error bars show the 95% confidence interval. 183
Figure 7.46: Psychovisual test results for 8nn connectivity using the AF filtering structure and the area attribute. The error bars show the 95% confidence interval. 183
Figure 7.47: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the area attribute. The error bars show the 95% confidence interval. 184
Figure 7.48: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the area attribute. The error bars show the 95% confidence interval. 184
Figure 7.49: Psychovisual test results for 4nn connectivity using the AF filtering structure and the power attribute. The error bars show the 95% confidence interval. 185
Figure 7.50: Psychovisual test results for 8nn connectivity using the AF filtering structure and the power attribute. The error bars show the 95% confidence interval. 185
Figure 7.51: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the power attribute. The error bars show the 95% confidence interval. 186
Figure 7.52: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the power attribute. The error bars show the 95% confidence interval. 186
Figure 7.53: The original 8bit greyscale, 720 x 576 Barbara image. 187
Figure 7.54: The Barbara image filtered using the AF filter structure, 4nn connectivity and the area attribute. 188
Figure 7.55: The Barbara image filtered using the AF filter structure, 4nn connectivity and the power attribute. 189
Figure 7.56: The setup of the second visual test using a slider bar to control the compression ratio/quality. 191
Figure 7.57: The input, filtered and reference images (8bit greyscale, 720 x 576) used in testing. 194
Figure 7.58: The best JPEG result output, showing the lowest compression ratio voted for, the average of those laying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for. 195
Figure 7.59: The input, filtered and reference images (8bit greyscale, 720 x 576) used in testing. 197

Figure 7.60: The best JPEG 2000 result output, showing the lowest compression ratio voted for, the average of those laying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for. 198
Figure 7.61: Psychovisual test results for 4nn connectivity using the AF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 200
Figure 7.62: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 200
Figure 7.63: Psychovisual test results for 8nn connectivity using the AF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 201
Figure 7.64: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 201
Figure 7.65: Psychovisual test results for 4nn connectivity using the AF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 202
Figure 7.66: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 202
Figure 7.67: Psychovisual test results for 8nn connectivity using the AF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 203
Figure 7.68: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results. 203
Figure 7.69: Input image, 720 x 576, 8bit greyscale Blackboard image, used to evaluate the visually lossless attribute values. 204
Figure 7.70: Filtered images used for testing using the area attribute, 4nn connectivity and the AF filter structure. 205
Figure 7.71: Filtered images used for testing using the power attribute, 4nn connectivity and the AF filter structure. 206
Figure 7.72: The input, filtered and reference image used in testing. 210
Figure 7.73: The best JPEG result output, showing the lowest compression ratio voted for, the average of those lying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for 211
Figure 7.74: The input, filtered and reference image used in testing. 212
Figure 7.75: The best JPEG 2000 result output, showing the lowest compression ratio voted for, the average of those lying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for. 213
Figure 8.1: Example to show how much of the image (Barbara 2 at 8bit greyscale, 720 x 576 pixels) is actually affected. 216
Figure 8.2:Example of how regions are extracted and labelled. 217
Figure 8.3: PSNR (dB) against compression ratio using the intermediate methods. 218
Figure 8.4: Example of intermediate filtered images using the 8bit greyscale 720 x 576 Barbara 2 image. 219
Figure 8.5: Example of intermediate filtered images using the 8bit greyscale, 720 x 576 Barbara 2 image. 220

Figure 8.6: The 8bit greyscale, 720 x 576 Barbara 2 image used as the input for intermediate morphology filters shown in Figure 8.7. 221
Figure 8.7: Region extraction using the morphological gradient. 222
Figure 8.8: Intermediate morphological results for an compressing the Barbara 2 image after filtering using an area size of 100. 223
Figure 8.9: Example to show the result of the gradient intermediate method using the 8bit greyscale, 720 x 576 Barbara 2 image. 224
Figure 8.10: Sample using the 8bit greyscale, 720 x 576 Barbara 2 image to show the regions extracted by the distance method. 225
Figure 8.11: Performance of the intermediate morphology distance method. 226
Figure 8.12: Sample images to show the result of the gradient intermediate method. 227
Figure 8.13: PSNR (dB) against compression ratio to show the performance of the full intermediate method. 228
Figure 8.14: Sample images to show the result of the full intermediate method on the 8bit greyscale, 720 x 576 Barbara 2 image. 229
Figure 9.1: Gaussian filter performance. 232
Figure 9.2: Original and corrupted of frame 10 of the 8bit greyscale, 353 x 288 Foreman sequence. 233
Figure 9.3: Gaussian filtering of frame 10 of the Foreman sequence. 234
Figure 9.4: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence. 235
Figure 9.5: Gaussian filtering of frame 100 of the Foreman sequence. 236
Figure 9.6: K-NN filter performance. 238
Figure 9.7: Original and corrupted of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 239
Figure 9.8: K-NN filtering of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence 240
Figure 9.9: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence. 241
Figure 9.10: K-NNfiltering of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence 242
Figure 9.11: Original and corrupted frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 244
Figure 9.12: Morphological filtering of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 245
Figure 9.13: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence. 246
Figure 9.14: Morphological filtering of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence. 247
Figure 9.15: Non-overlapping spatio-temporal filtering. 250
Figure 9.16: Extension of the 4nn (left) and 8nn (right) to 3D giving the 6nn (or 4_{nn}^{3D}) and 26nn (or 8_{nn}^{3D}) respectively. 251
Figure 9.17: Overlapping window method. 256
Figure 9.18: Combined spatial and spatio-temporal area attribute. 263
Figure 9.19: Combined spatial and spatio-temporal power limit. 264
Figure 9.20: The original, corrupted (uncompressed) and Gaussian frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 280
Figure 9.21: Simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 281
Figure 9.22: The original, corrupted (uncompressed) and Gaussian frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence. 282

Figure 9.23: Simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence. 283
Figure 9.24: The original and corrupted frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence compressed to 20:1 (0.4bpp or 990Kbps) using the H263 CODEC. 285
Figure 9.25: Compression of the simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence to a ratio of 20:1 (0.4bpp or 990Kbps). 286
Figure 9.26: The original and corrupted frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence compressed to 20:1 (0.4bpp or 990Kbps) using the H263 CODEC. 287
Figure 9.27: Compression of the simultaneous 2D and 3D filtering using frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence to a ratio of 20:1 (0.4bpp or 990Kbps). 288
Figure 9.28: Noise variance against filter values. 291
Figure 9.29: Frequency of the best block size and percentages of blocks to use. 293
Figure 9.30: Frequency of the best block size and percentages of blocks to use. 294
Figure 9.31: Original, corrupted and Gaussian filtered frame 5 of the 352 x 288 8bit greyscale Stefan sequence. 296
Figure 9.32: Frame 5 of the 352 x 288 8bit greyscale Stefan sequence filtered using the K-NN and morphological methods. 297
Figure 9.33: Original, corrupted and Gaussian filtered frame 45 of the 352 x 288 8bit greyscale Stefan sequence. 298
Figure 9.34: Frame 45 of the 352 x 288 8bit greyscale Stefan sequence filtered using the K-NN and morphological methods. 299
Figure 12.1: Venn diagram of two basic sets. 310
Figure 12.2: Venn diagram of subsets. 310
Figure 12.3: Venn diagram of the intersection of two sets. 312
Figure 12.4: The compliment of a set. 313
Figure 12.5: The set difference. 313
Figure 12.6: Translation of a set. 314
Figure 12.7: Reflection of a set. 315
Figure 13.1: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale. 317
Figure 13.2: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale. 318
Figure 13.3: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale. 318
Figure 13.4: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale. 319
Figure 13.5: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale. 319
Figure 13.6: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale. 320
Figure 13.7: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale. 320

Figure 13.8: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale. 321
Figure 13.9: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale. 321
Figure 13.10: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale. 322
Figure 13.11: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale. 322
Figure 13.12: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale. 323
Figure 13.13: Area and contrast attribute results for the Barbara 2 image with 14dB of noise. 325
Figure 13.14: Volume and power attribute results for the Barbara 2 image with 14dB of noise....326
Figure 13.15: Area and contrast attribute results for the Barbara 2 image with 21dB of noise. 327
Figure 13.16: Volume and power attribute results for the Barbara 2 image with 21dB of noise....328
Figure 13.17: Area and contrast attribute results for the Barbara 2 image with 28dB of noise. 329
Figure 13.18: Volume and power attribute results for the Barbara 2 image with 28dB of noise....330
Figure 13.19: Area and contrast attribute results for the Barbara 2 image with 35dB of noise. 331
Figure 13.20: Volume and power attribute results for the Barbara 2 image with 35dB of noise....332
Figure 13.21: Area and contrast attribute results for the Boats image with 14dB of noise....333
Figure 13.22: Volume and power attribute results for the Boats image with 14dB of noise. 334
Figure 13.23: Area and contrast attribute results for the Boats image with 21dB of noise....335
igure 13.24: Volume and power attribute results for the Boats image with 21dB of noise. 336
Figure 13.25: Area and contrast attribute results for the Boats image with 28dB of noise....337
Figure 13.26: Volume and power attribute results for the Boats image with 28dB of noise. 338
Figure 13.27: Area and contrast attribute results for the Boats image with 35dB of noise....339
Figure 13.28: Volume and power attribute results for the Boats image with 35dB of noise. 340
Figure 13.29: Area and contrast attribute results for the Goldhill image with 14dB of noise. 341
Figure 13.30: Volume and power attribute results for the Goldhill image with 14dB of noise. 342
Figure 13.31: Area and contrast attribute results for the Goldhill image with 21dB of noise. 343
Figure 13.32: Volume and power attribute results for the Goldhill image with 21dB of noise. 344
Figure 13.33: Area and contrast attribute results for the Goldhill image with 28dB of noise. 345
Figure 13.34: Volume and power attribute results for the Goldhill image with 28dB of noise. 346
Figure 13.35: Area and contrast attribute results for the Goldhill image with 35dB of noise. 347
Figure 13.36: Volume and power attribute results for the Goldhill image with 35dB of noise. 348
Figure 13.37: JPEG rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the AF filter structure. 357
Figure 13.38: JPEG rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the ASF filter structure. 358
Figure 13.39: JPEG rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the AF filter structure.. 359
Figure 13.40: JPEG rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the ASF filter structure. 360

Figure 13.41: JPEG rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the AF filter structure. 361
Figure 13.42: JPEG rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the ASF filter structure. 362
Figure 13.43: JPEG 2000 rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the AF filter structure. 363
Figure 13.44: JPEG 2000 rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the ASF filter structure. 364
Figure 13.45: JPEG 2000 rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the AF filter structure. 365
Figure 13.46: JPEG 2000 rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the ASF filter structure. 366
Figure 13.47: JPEG 2000 rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the AF filter structure. 367
Figure 13.48: JPEG 2000 rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the ASF filter structure. 368

List of Tables

Table 3.1: The use of objective measurements. 24
Table 3.2: Five-point scoring scale. 27
Table 3.3: The comparison scale. 28
Table 6.1: Number of complete scans to reconstruct the image. Shaded cells show best results. 90
Table 6.2: Average execution times (mS) for the different reconstruction processes taken over 76 8bit greyscale image of size 720 x 576 pixels. Shaded cells show the best results. 90
Table 6.3: Average execution time (in seconds) for an area size of 47 taken over 50 8bit greyscale images with a size of 720 x 576 pixels. 115
Table 7.1: The optimum sigma values and mask sizes for noise removal using the Gaussian filter and their performance gain in PSNR. 121
Table 7.2: PSNR values for the output of the JPEG and JPEG 2000 CODEC's using the optimum Gaussian filtered images as the input. The CODEC outputs are compared to the original noise free image. 125
Table 7.3: The optimum attribute values for noise removal and their performance gain in PSNR. The shaded values show the best outputs and the outlined show the worst. 148
Table 7.4: Comparison of the best Gaussian and Morphological noise reduction results. Shaded cells show the best results for the given image. 154
Table 7.5: Output PSNR of the optimum attribute values using JPEG at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst results. ..156
Table 7.6: Output PSNR of the optimum attribute values using JPEG 2000 at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst results. ..157
Table 7.7: Average error in the noise estimation with various amounts of noise. 161
Table 7.8: Comparison between estimated and true noise variance measurements for the 8bit greyscale, 720 x 576 Blackboard and Girl images. 163
Table 7.9: Estimated Gaussian filter variable values for the 8bit greyscale, 720 x 576 Blackboard and Girl images. 164
Table 7.10: Estimated morphological filter variable values and the resultant PSNR for the 8bit greyscale Blackboard and Girl images. Shaded cells show the best output for that particular image. 165
Table 7.11: JPEG and JPEG 2000 compression of the Gaussian filtered Blackboard and Girl image (8 bit greyscale at 720 x 576 pixels) using a compression ratio of 20:1 (0.4bpp). 171
Table 7.12: Output PSNR of the morphological filters using unknown data with JPEG at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst reults. 173
Table 7.13: Output PSNR of the morphological filters using unknown data with JPEG 2000 at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst reults 174
Table 7.14: Attribute limits used for the test. 181
Table 7.15: Attribute values that were determined to be visually lossless. 187

Table 7.16: Results to show how many observers thought what compression ratios compared to the filtered at 20:1 (0.4bpp). The shaded results show the best outputs. 192
Table 7.17: The average compression ratio laying within the most frequently chosen range of Table 7.16. 192
Table 7.18: Attribute limits used for the test. 199
Table 7.19: Attribute values that were determined to be visually lossless. 199
Table 7.20: Results to show how many observers thought what compression ratios compared to the filtered at 20:1 (0.4bpp). The shaded results show the best outputs. 207
Table 7.21: The average compression ratio lying within the most frequently chosen range of Table 7.20. 208
Table 9.1: Optimum filter outputs using the Gaussian filtering method....231
Table 9.2: Optimum attribute values for spatial processing of video. Shaded cells show the best results, outlined cells show the worst results. 243
Table 9.3: CODEC PSNR output at a compression ratio of 20:1 (0.4bpp or 990Kbps) using Gaussian and K-NN preprocessed sequences. Shaded cells show the best result for that particular configuration. 248
Table 9.4: CODEC PSNR output at a compression ratio of 20:1 (0.4bpp or 990Kbps) using morphologically preprocessed sequences. Shaded cells show the best result for that particular configuration and the outlined cells show the worst output....249
Table 9.5: Optimum attribute values and resultant PSNR for non-overlapping method. Shaded cells show the best result for that particular configuration and the outlined cells show the worst output.....252
Table 9.6: CODEC PSNR output at 20:1 (0.4bpp) using non-overlapping preprocessing. Shaded cells show the best outputs and the outlined cells show the worst outputs. 254
Table 9.7: Optimum attribute values and resultant PSNR for overlapping method. Shaded cells show best outputs and outlined cells show the wrost. 257
Table 9.8: CODEC PSNR output at 20:1 (0.4bpp or 990Kbps) using overlapping preprocessing for the Foreman sequence corrupted with noise to 28.14dB. Shaded cells show the best results and the outlined cells show the worst....259
Table 9.9: CODEC PSNR output at 20:1 (0.4bpp or 990Kbps) using overlapping preprocessing for the Kiel sequence corrupted with noise to 28.12dB. Shaded cells show the best results and the outlined cells show the worst....260
Table 9.10: Optimum results for the combined spatio-temporal methods. Shaded cells show the best results and the outlined cells show the worst....265
Table 9.11: Results for the Foreman sequence using MJPEG and MPEG-II compression. Shaded cells show the best results and the outlined cells show the worst. 267
Table 9.12: Results for he Foreman sequence using BWV and H263 compression. Shaded cells show the best results and the outlined cells show the worst. 268
Table 9.13: Results for the Kiel sequence using MJPEG and MPEG-II compression. Shaded cells show the best results and the outlined cells show the worst....269
Table 9.14: Results for the Kiel sequence using BWV and H263 compression. Shaded cells show the best results and the outlined cells show the worst....270

Table 9.15: Optimum results for the simultaneous spatio-temporal method. Shaded cells show the best results and the outlined cells show the worst. 274
Table 9.16: Simultaneous results for the Foreman sequence using the MJPEG and MPEG-II CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst. 275
Table 9.17: Simultaneous results for the Foreman sequence using the BWV and H263 CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst. 276
Table 9.18: Simultaneous results for the Kiel sequence for the MJPEG and MPEG-II CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst. 277
Table 9.19: Simultaneous results for the Kiel sequence for the BWV and H263 CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst. 278
Table 9.20: Estimated filter values to use and the resultant PSNR after filtering 295
Table 9.21: PSNR of the Stefan sequence after compression to a ratio of 20:1 (0.4bpp), with the best output for each CODEC shaded, not including the unfiltered sequences. 300
Table 13.1: The optimum sigma values and mask sizes for noise removal using the Gaussian filter and their performance gain in PSNR. 323
Table 13.2: PSNR values for the output of the JPEG and JPEG 2000 CODECs using the optimum Gaussian filtered images as the input. The CODEC outputs are compared to the original noise free image. 324
Table 13.3: The optimum attribute values for noise removal and their performance gain in PSNR for noise levels of 14dB and 21dB. Best results are shaded and the worst are outlined. 349
Table 13.4: The optimum attribute values for noise removal and their performance gain in PSNR for noise levels of 28dB and 35dB. Best results are shaded and the worst are outlined. 350
Table 13.5: Output PSNR of the optimum attribute values using JPEG for 14dB and 21dB of noise. Best results are shaded and the worst are outlined 351
Table 13.6: Output PSNR of the optimum attribute values using JPEG for 28dB and 35dB of noise. Best results are shaded and the worst are outlined 352
Table 13.7: Output PSNR of the optimum attribute values using JPEG 2000 for 14dB and 21dB of noise. 353
Table 13.8: Output PSNR of the optimum attribute values using JPEG 2000 for 28dB and 35dB of noise. 354

List of Acronyms

AC	Area Closing
ACO	Area Close Open
ADC	Analogue to Digital Converter
AF	Alternating Filter
AI	Artificial Intelligence
AM	Area Morphology
AO	Area Opening
AOC	Area Open Close
ASF	Alternating Sequential Filter
ASIC	Application Specific Integrated Circuit
AWGN	Additive White Gaussian Noise
bpp	bits per pixel
BWV	Bath Wavelet Video
CCD	Charge Coupled Device
CGI	Computer Generated Imagery
CIE	Commission Internationale de l'Eclairage
CIF	Common Image Format
CODEC	Coder/Decoder
CRT	Cathode Ray Tube
DAC	Digital to Analogue Converter
DCT	Discrete Cosine Transform
DSCQS	Double Stimulus, Continuous Quality Scale
DSCS	Double Stimulus, Comparison Scale
DSIS	Double Stimulus, Impairment Scale
DVC	Digital Video Camera
DVD	Digital Versatile Disc
DWT	Discrete Wavelet Transform
ELIC	Embedded Lossless Image Coder
ENG	Electronic News Gathering
EZW	Embedded Zerotree Wavelet
FIFO	First In, First Out
FPGA	Field Programmable Gate Array
fps	Frames per second
FSE	Flat Structuring Element
GOP	Group of Pictures
HDTV	High Definition Television
HVS	Human Visual System
ISI	Inter Stimulus Interval

ISO	International Standards Organisation
ITU	International Telecommunications Union
JND	Just Noticeable Difference
JPEG	Joint Photographics Experts Group
Kbps	Kilo-bits per second
K-NN	K Nearest Neighbours
LMS	Least Mean Squares
MAD	Mean Absolute Difference
Mbps	Mega-bits per second
MIC	Morphological Image Cleaning
MJPEG	Motion-JPEG
MOS	Mean Opinion Score
MPEG	Moving Pictures Experts Group
MRI	Magnetic Resonance Imaging
MSAM	Morphological Signal Adaptive Median
MSE	Mean Square Error
NTSC	National Television Systems Committee
OFDM	Orthogonal Frequency Division Multiplexing
PAL	Phase Alternating Line
PCB	Printed Circuit Board
PCM	Pulse Code Modulation
PDF	Probability Density Function
PSNR	Peak Signal to Noise Ratio
PVR	Personal Video Recorder
QCIF	Quarter Common Image Format
QPSK	Quadrature Phase Shift Keying
RGB	Red, Green and Blue
RLE	Run Length Encoding
RMSE	Root Mean Square Error
SAD	Sum of Absolute Differences
SAR	Synthetic Aperture Radar
SE	Structuring Element
SHD	Super High Definition
SNR	Signal to Noise Ratio
SS	Single Stimulus
SSMR	Single Stimulus, Multiple Repetition
STL	Standard Template Library
VLC	Variable Length Coding

1 Introduction

This thesis details the development of preprocessing techniques using Mathematical Morphology to improve the compressibility of digital images and video. Two processes achieve this, the first process removes noise, from the image sequence, which saves bits that would otherwise be taken away from the image data. Furthermore, as noise is uncorrelated and high frequency, it does not compress well. Secondly, the complexity of the image sequences can be reduced by removing image features and details, providing this is done in such a way that the perceived visual quality of the sequence is not reduced., The aim of this work is to create a stand-alone preprocessing system that will work on a variety of systems such as Moving Picture Experts Group (MPEG) and wavelet based *coders/decoders* (CODEC's). All work contained within this report is based upon digital images and video. Before proceeding with the work covered, (see chapters 6, 7, 8 and 9), the basic principles behind digital video (see chapter 2) and preprocessing (see chapter 5) are introduced. These show why preprocessing and noise reduction are used and the benefits of digital video.

Chapter 4 provides an introduction to mathematical morphology starting with the basic operations of erosion and dilation. This is then extended to form more complex operators such as openings and closings. Chapter 6 then looks at the efficient implementation of some of the morphological operators, namely reconstruction, Area Morphology (AM) and Attribute Morphology. Current implementation methods are evaluated and a new efficient method is developed for use with the Alternating Sequential Filter (ASF), which significantly increases the performance of this filter compared to all of the current methods. In addition, a new attribute, the power, is proposed to form a closer link with the Human Visual System (HVS). Chapter 7 develops these ideas into two preprocessing systems for images. The first system evaluates the use of attribute morphology, both in the Alternating Filter (AF) and ASF filter structures, for reducing the effects of Gaussian noise on an image. The area, contrast, volume and power attributes are used, as are various levels of noise and connectivity. The second system is based on a psychovisually lossless measure. Uncorrupted input images are used and filtered using both the AF and ASF filter structures, area, contrast, volume and power attributes, and 4nn and 8nn connectivity. Psychovisual evaluations are then used to determine visually lossless attribute values for each filter combination. The results allow images to be filtered without an observer being able to detect this. Since this has reduced the information within the image, a CODEC can compress the image more efficiently. To ensure this is the case, images have also been compressed using the attribute values found and several compression schemes. The resultant images were placed in a second psychovisual evaluation, which proves that an observer cannot tell the difference. In addition this also proved that the resultant images are more compressible,

Chapter 9 develops the noise reduction methods of chapter 7 into a spatio-temporal system. The area and power attributes are used in conjunction with the AF and ASF filter structures and both 4nn and 8nn connectivity. A full spatio-temporal system is then developed in several stages. Initially the spatial method is simply applied to each frame, which is expanded to process an entire block of a

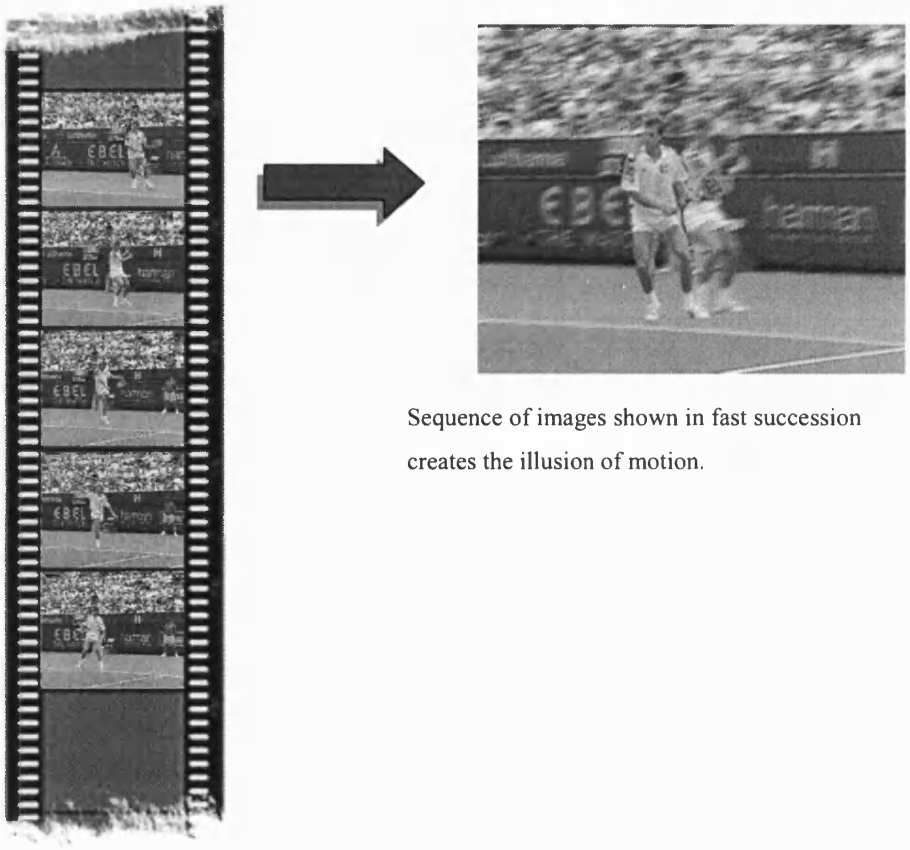
video sequence. This is then expanded further to form a sliding window and merged with the spatial method to form a combined and simultaneous spatial and spatio-temporal preprocessing system. Whilst spatio-temporal filtering is common for other image processing methods, mathematical morphology has not been applied in this way to digital video before. Finally conclusions are drawn giving details of what has been accomplished and possible improvements to the systems developed are also outlined.

2 Introduction to Video

Video in any form is essentially a series of pictures shown in rapid succession (see Figure 2.1). Showing these pictures in this fashion creates the illusion of motion. There are a variety of methods in which these sequences can be stored and played back. This chapter gives a brief overview of some of these methods and details the advantages that digital systems have over conventional analogue systems.

2.1 History of Imaging

Video is simply a series of still photographs. Motion Pictures and television has been around for over a century [1], [2]. The birth of photography can be traced back to the Camera Obscura (Latin for dark room), which has been around since the time of *Aristotle* although the first recorded usage was not until *Leonardo da Vinci* used it in 1519AD to aid his drawings. In 1824AD, *Peter Roget* wrote an article, *The Persistence of Vision with Regard to Moving Objects*, which proved that the human eye retains an image for a fraction of a second after it has been shown. It was not until 1839AD that the phrase '**Photography**' (from the Greek for light and writing) by *Sir John Herschel* was used.



Sequence of images shown in fast succession creates the illusion of motion.

Figure 2.1: The illusion of motion.

Video was originally shown by projecting photographic film onto a screen, in much the same way as cinema projection systems work today. Numerous inventors had the vision of changing this by somehow scanning and transmitting pictures. Television has been independently developed across the world. For example, *John Logie Baird* is credited with the birth of television in the UK and *Jenkins* in the USA. John Logie Baird was however the first to exploit television commercially with his company called '**Baird Television**', which started life as Television Ltd [1] - [4]. The system used an electromechanical device, the '**Nipkow disc**' invented in 1884AD by a German named *Paul Gattlieb Nipkow*, to scan images for transmission and a similar device to display them. This was a major achievement of the time, although there were problems with the system. The device scanned images vertically, from bottom to top across the image from right to left, rather than horizontally. Initially the system used 30 vertical lines with a frame rate of 12.5 frames per second, which resulted in flickering. Also, unlike modern television, the Baird system used an aspect ratio of 3:7, giving a vertical widescreen format. By 1936 the system had been improved to cope with 25 frames per second and 240 lines.

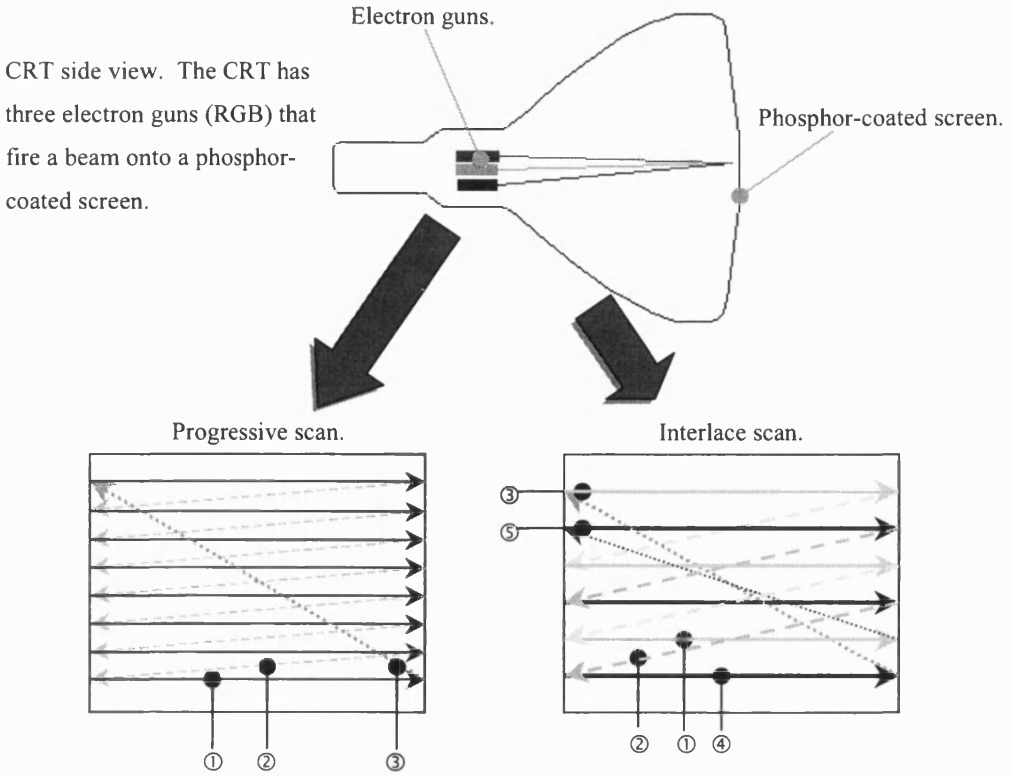
The electromechanical system eventually gave way to a fully electronic system devised by '**EMI Ltd**' in the 1930's [1], [2]. However, this type of system was proposed by a Scottish engineer named *A.A. Campbell Swinton* in 1908 [5]. This was the first system to use the Cathode Ray Tube (CRT) and the 50 fields per second system. Regular transmissions started in 1936 using this system. The BBC provided the first regular transmissions of entertainment programmes using high definition images, 405 lines, using systems developed by British teams from EMI and Marconi. With the exception of two major improvements, the fundamental principles of television have remained unchanged until the introduction of digital signal representation. The 625-line system was introduced in 1964 and is still in use today, although the 405-line system was still used until 1985. A second major improvement came in 1969 with the advent of colour television, although America had introduced its colour system in 1953 [6]. Despite its late arrival, colour television had been demonstrated working by Baird in 1944 [2].

Since the introduction of digital, television has seen several improvements such as interactive media, High Definition Television (HDTV) and Super High Definition (SHD) have been demonstrated [6], [7]. HDTV has been designed specifically for the use of digital television and is leading the way to the adoption of the widescreen aspect ratio, 16:9, for television sets. The first video recordings were made by Baird on a Gramophone Videodisc, called '**Phonovision**', between 1927 and 1935. A more practical and economical solution did not appear until the 1960's with the introduction of the videotape, and more recently, the introduction of the Laser Disc. However, these are analogue solutions. The introduction of digital television in the 1990's required a more versatile solution. Thus the Digital Versatile Disc (DVD) was introduced which can hold up to 10 hours of D1 video, which is 720 x 576 pixels at 25 frames per second (fps) with 24bpp, video using MPEG-II compression in addition to multiple channel and multi-lingual audio. More recently, the Personal Video Recorder (PVR) has been introduced, which records onto a hard disc.

2.2 Analogue Video

The most common method of recording video throughout the late 19th and 20th centuries was use of celluloid film. This process uses a camera to expose several frames of photographic film every second and is still used today by the movie industry. However, this method cannot be broadcast at its original resolution and copies must be made and distributed for multiple audiences. Analogue television, which was first introduced in the 1930's, has been used to broadcast to a wide audience. This uses a signal with an infinitely variable parameter (i.e. voltage or frequency) to broadcast the video. To represent a sequence at its original resolution, known as broadcast quality, a relatively large bandwidth is required. In order to reduce the bandwidth, the resolution of the image is reduced. Using this technique, the signals can be broadcast allowing a large audience to view the same video source at the same time. However, because of the analogue modulation techniques used, it is very susceptible to noise, which can be hard to remove or detect.

Until recently, the most common and economical display system has been the CRT, invented by a German scientist, *Karl Ferdinand Braun*, in 1897. This works by scanning an electron beam across a screen that has been coated with phosphors (see Figure 2.2). Individual elements are then illuminated at different intensities across the line. Once the beam reaches the end of the line, it is turned off and moved back and down to the start of the next line, a process known as fly back. The whole process is then repeated until the entire display has been scanned. Each frame is displayed in the same way, with 25 frames being shown every second in the UK standard. This method of display is called progressive. The frame rates and line spacing used within television are based on psychophysical experiments. Early in the development of television, flickering became a problem. To overcome this, interlacing was introduced which works by reducing the horizontal resolution by a factor of a half and doubling the frame rate so the amount of information is transmitted remains constant. The image is displayed in a similar way to progressive displays as shown in Figure 2.2. The beam is scanned across the display as before, but instead of moving down one line, it moves down two lines, in effect only scanning the even lines. Once the beam has finished scanning the display, the beam then scans the alternate, odd, lines that were not scanned before. This gives the illusion of having the same amount of horizontal information. The complete scanning of one set of lines, odd or even, is referred to as a field and the complete scanning of both odd and even lines is still referred to as a frame. Most of the motion captured by a camera moves across the frame rather than up or down it. This is the main reason the display system scans across and down.



① The beam is first scanned across the screen.
② The beam is turned off and moved to the next line. This is called the flyback.
③After the bottom scan has been done, the beam is moved back to the top and the process started over again.

①,④ The beam is first scanned across the screen, starting with even lines.
② The beam is turned off and moved down, skipping the next line. This is called the flyback.
③,⑤ After the bottom scan has been done, the beam is moved back to the top and the process started over again on the odd lines (if the even lines have just been processed).

Figure 2.2: Progressive and interlaced scanning.

2.2.1 Analogue Colour Representation

Using the tristimulus theory, meaning that the HVS uses the Red, Green and Blue (RGB) colour representation, the most obvious colour space to use is RGB. However, the first television broadcasts and receivers worked only in black and white using luminance, which can easily be derived from RGB as shown in equation 2.1. Because the HVS is less sensitive to colour and to keep backwards compatibility, the YUV system is used to represent video. It should be noted however, that there are alternate representations such as YIQ as used by the American National Television Systems Committee (NTSC) system [6]. The luminance is represented by Y and the colour (or chrominance) is represented by the colour difference signals U and V as given in equations 2.2 and 2.3. This allows

black and white receivers to view colour transmissions. The bandwidth can also be reduced by decreasing the horizontal resolution of the chrominance information by a factor of two. In addition the bandwidth requirements can be reduced even further by using a quadrature modulator to produce a composite signal. This relationship can be seen more clearly in Figure 2.3. With respect to the original progressive RGB given by the camera, the interlaced composite output is a sixth of the size.

$$Y = 0.299R + 0.587G + 0.114B \tag{2.1}$$

$$U = R - Y = 0.701R - 0.587G - 0.114B \tag{2.2}$$

$$V = B - Y = 0.299R - 0.587G + 0.886B \tag{2.3}$$

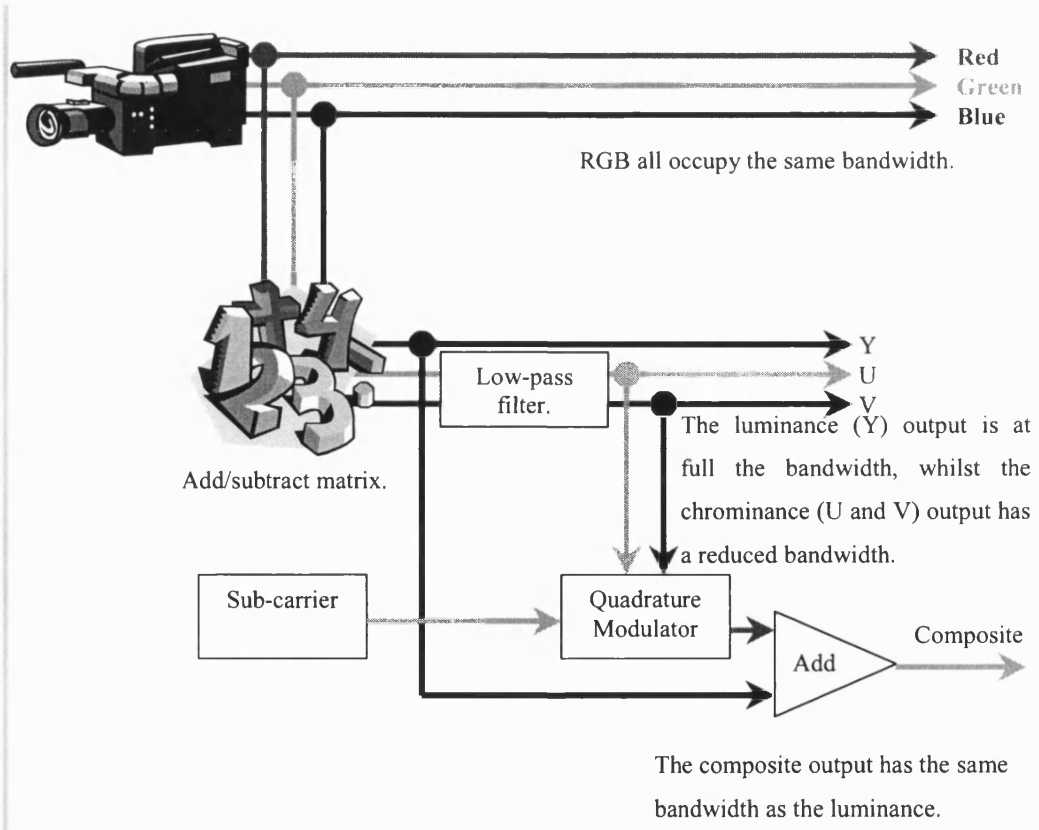


Figure 2.3: The relationship between the three primary colour representations used in the analogue video system.

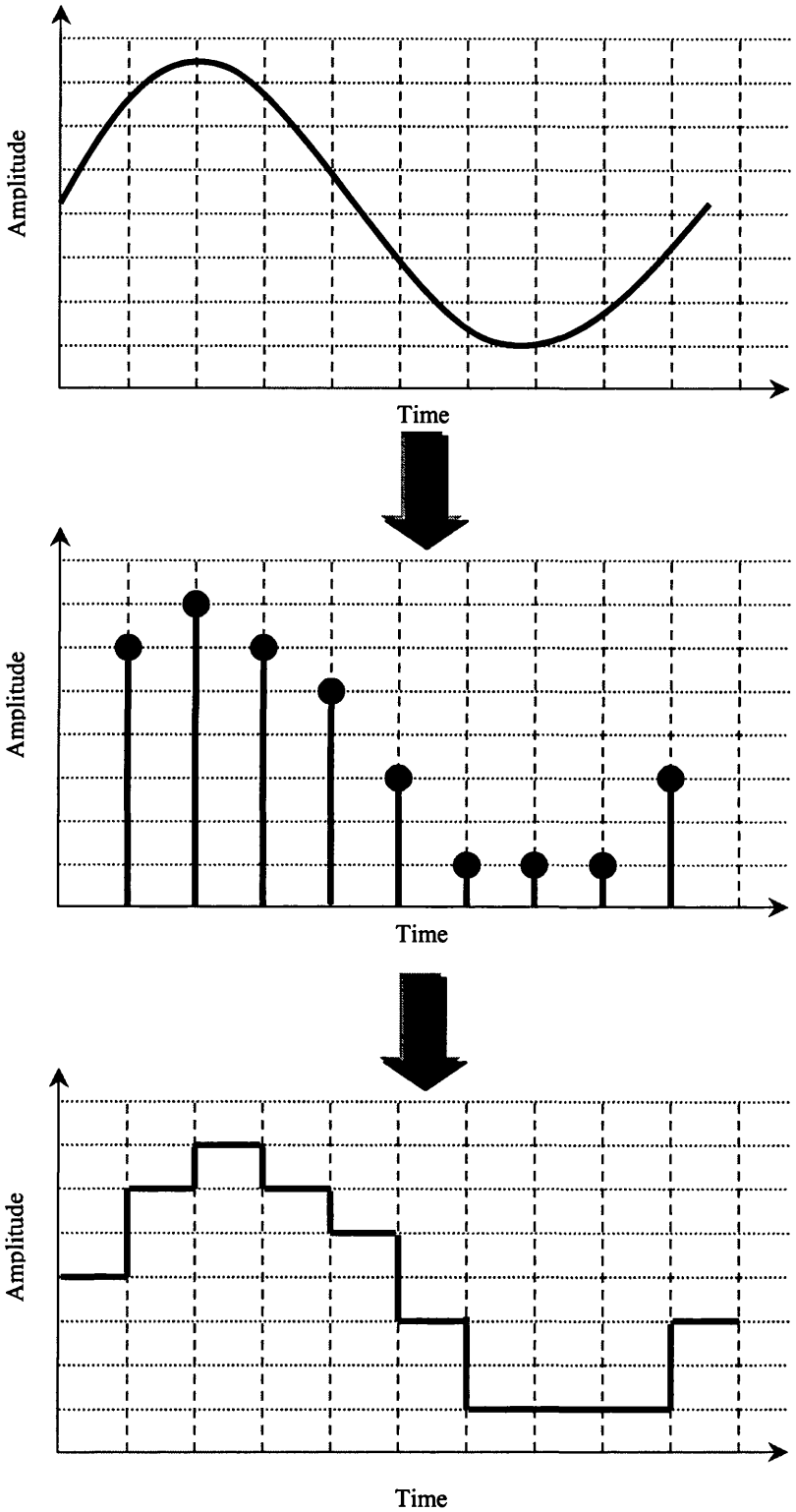
2.3 Digital Video

The digital format is simply a different way of representing video information [8] - [11]. All digital formats work by converting analogue signals into digital information. The only difference between digital systems is the stage at which this conversion is done. For example, Charge Coupled Device's (CCDs), such as those found in most Digital Video Cameras (DVC), usually have Analogue to Digital Converters (ADCs) integrated into them. The analogue signal is converted to a digital signal by sampling it at given intervals and then quantising the samples into discrete values, as shown in Figure 2.4. This is method known as Pulse Code Modulation (PCM), the result is a series of discrete samples known as pixels which when displayed correctly form an approximation of the original video signal. The entire image is sampled in this way and the resulting samples converted into a binary format. A binary signal can then be coded using error correction methods such as hamming and convolution codes, which allows the digital format to be more robust against noise [12]. Various transmission techniques allow a further reduction in error and can use less power to cover the same area as an analogue transmission. There are several key points to note about a digital system:

- Quality does not change with transmission medium,
- The video data can be copied with little or no loss,
- Are cheaper to produce equipment (transmission, receiver and storage),
- Easier to perform post-production on (i.e. special effects),
- Extends the playback time of storage devices,
- Allows miniaturisation.

These factors make digital systems more advantageous than analogue ones. For example, miniaturising equipment has allowed Electronic News Gathering (ENG) to be accomplished with less equipment and staff reducing costs for the news company. However, in most current display devices, such as the CRT, the data is converted back into an analogue format using a Digital to Analogue Converter (DAC) in order to display it. More recently, digital video has started to appear in larger applications. For example, *George Lucas* filmed all of the "**Starwars II:Attack of the Clones**" film using only high definition DVC's. In addition, digital cinemas have started to be tested and will inevitably be rolled out into the mainstream, which has the obvious advantage over film in that copies do not need to be made. This could also mean the possibility of releasing movies simultaneously world-wide instead of country by country. The film could be simply downloaded via the Internet or satellite connection to the cinema. There are numerous other uses for digital video:

- Video conferencing,
- Video On Demand,
- Interactive Media,
- PVR,
- Telemedicine,
- HDTV [6], [7],
- Digital cinema.



The analogue input signal

The amplitude of the signal is sampled periodically to form a discrete time signal, which is quantised to form a digital signal.

The sampled values can then be processed and turned back into an analogue signal.

Figure 2.4: Analogue to digital signal conversion.

2.3.1 Bandwidth of Analogue and Digital Video

To show why compression is needed, consider a standard analogue Phase Alternating Line (PAL) transmission as used in the UK. This consists of 25fps or 50 fields per second. Each frame consists of 625 lines, which give a line rate of:

$$f_h = 625 \times 25 = 15{,}625Hz \tag{2.4}$$

Each line contains 720 pixels, which are visible, but actually contains 864 pixels for the luminance and 432 pixels for the chrominance, which includes other data such as teletext, synchronisation and time. This is made from 432 cycles for the luminance and 216 cycles for the chrominance of video waveform. This then gives the bandwidth of the analogue signal, for the luminance and chrominance signals as:

$$BW_{luminance} = 15{,}625Hz \times 432 = 6.75MHz \tag{2.5}$$

$$BW_{chromanance} = 15{,}625Hz \times 216 = 3.375MHz \tag{2.6}$$

Using the standards for digital television, the luminance channel is sampled at 13.5MHz and the two chrominance channels need to be sampled at 6.75MHz each [13]. This is comes from the Nyquist sampling theorem, which states that to avoid aliasing, a signal should be sampled at twice the highest frequency the input signal can be, which gives a total sampling rate of 27MHz. Most studios will use 10 bits to represent this information giving a data rate of 270Mbits per second. Consumer equipment uses only use 8 bits, which would require 216Mbits per second. Both methods will require a bandwidth in excess of 100MHz, which does not compare well with the 5.5MHz used in the analogue system [6], [14]. Hence the digital information must be compressed. The transmission format used can easily be changed to reduce the bandwidth, for example Quadrature Phase Shift Keying (QPSK) or Orthogonal Frequency Division Multiplexing (OFDM) can be used. However, using image and video compression techniques can make a greater reduction.

2.3.2 Sources of Redundancy in Video

Most images and video sequences, both analogue and digital have a large amount of redundant data that can be removed in order to reduce the data rate. There are four types of redundancy that can be found in images:

- Spatial/Intra Redundancy,

  Some areas of an image will have the same colour or pattern (i.e. blue sky) as shown in Figure 2.5. This redundancy can be removed by saying that the entire block is made from one single colour. Thus only a small amount of data is required in order to do this, saving on sending repeated information. JPEG for example uses this method in order to achieve compression.

- Temporal/Inter-frame Redundancy [15], [16],

  Adjacent frames are usually correlated in some way which implies that there is some redundancy between the frames. Thus it is more efficient to just transmit the changes between frames instead of the whole image as shown in Figure 2.6. This is usually which is done by the use of motion compensation. This tells the receiver where objects have moved from with reference to the previous frame.

- Statistical Redundancy,

  After the data has been transformed into another domain and quantised, there may still be redundancy. Some symbols will inevitably occur more often than others. For example, in the English language, the letter *"E"* occurs more often than any other. Thus symbols, which occur more often, are assigned fewer bits than those that are less probable. This process is referred to as entropy encoding.

- Psychovisual Redundancy [17],

  Even if all of the spatial, temporal and statistical redundancy is removed from the sequence, there may still be a little redundancy left. Most images contain information that is not noticed by the HVS. For example, a large flat region is not usually composed of one value but several values averaging around the same point. Thus it is more efficient to make the region one value, resulting in less data for the CODEC to handle and in turn creating a more compressed output.

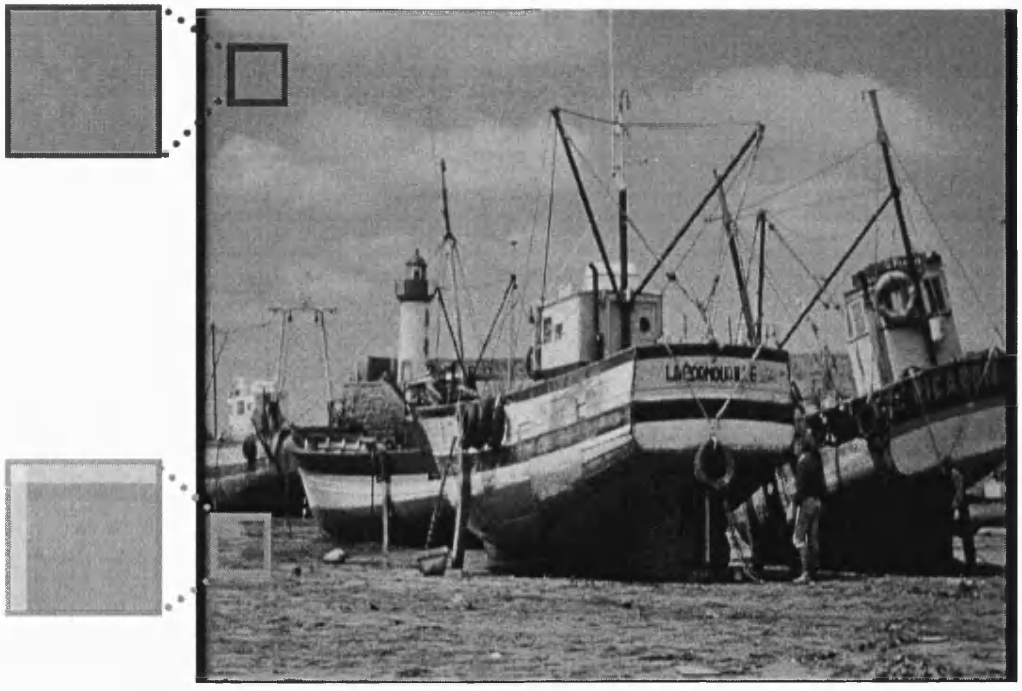
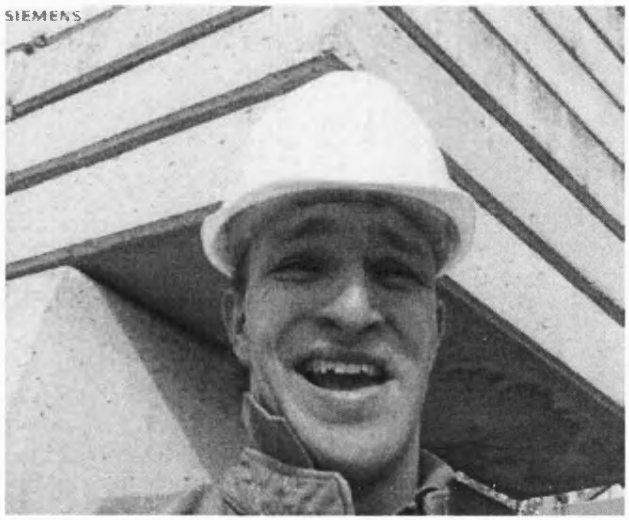
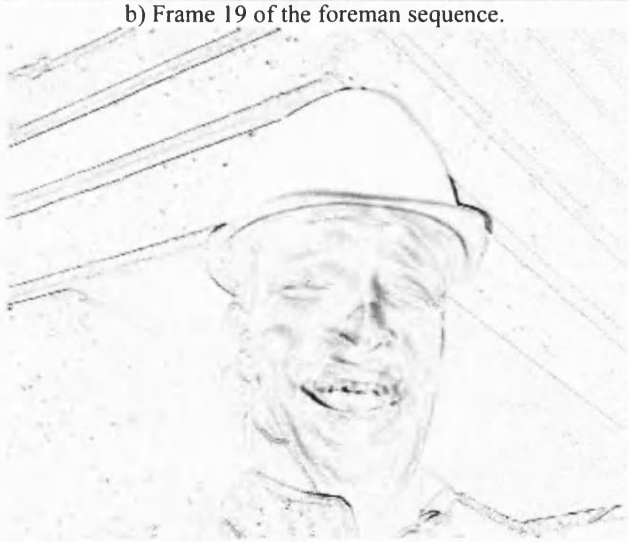


Figure 2.5: Spatial redundancy in images. The two blocks marked in the sky and on the sand are almost a constant colour and hence contain little information.



a) Frame 18 of the foreman sequence.

b) Frame 19 of the foreman sequence.



c) The difference between frames. Dark spots indicate a large change between the two frames.

Figure 2.6: The temporal redundancy of video.

2.3.3 Still Image Compression

There are many compression methods available, each with their own advantages and disadvantages. Any compression system will fall into one of two categories:

- Lossless,

  This type of system compresses and decompresses an image bit for bit. There is absolutely no loss in data. However, such systems do not compress to high compression ratios, typically only a 2:1 ratio is achieved.

- Lossy,

  This is by far the most commonly used type of system. It removes information from the image in order to achieve higher compression ratios. Typically the information removed will be redundant, but inevitably not all. A subset of lossy CODEC's is the **'visually lossless'** class, which although being lossy in nature, visually there is no loss in quality.

The encoding or compression stage is made-up of four components (see Figure 2.7):

- The transforms cannot distinguish between image features, noise and redundant information so preprocessing is used to remove psychovisually redundant information and noise, thus allowing the output of the transform to be quantised better.

- A transform is then used to convert the data into a more useful representation. The two main transforms currently in use are the Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT), although there are many more such as the Haar and fractal transforms.

  - The DCT method splits the image into blocks, usually 8 x 8 or 16 x 16 pixels. The quantiser and entropy coder then work on each block individually instead of processing the image as a whole. The DCT transform is used in many CODEC's such as the Joint Photographics Experts Group (JPEG) CODEC.

  - Instead of splitting an image into blocks of data, the DWT transforms the entire image. The resultant image is split into filtered images with the coefficients contained in the sub-bands, which are localised in both time and frequency. For example, the high frequency and low frequency components are separated. There are various different ways in which this can then be quantised [18], [19]. The structure of the wavelet filter bank is taken advantage of by embedded quantisation schemes such as the Embedded Zerotree Wavelet (EZW) [20], Embedded Lossless Image Coder (ELIC) [21], [22]. The transform stage compacts the image information into a few low frequency bands. New CODEC's, such as JPEG 2000, are taking advantage of the DWT.

- The quantiser is the part of the CODEC that actually compresses the data. Information from the transform is prioritised, with the lower priority information being discarded. As more information is discarded, the more the compression increases. There are many different quantisation schemes in use [23]. The method chosen for a CODEC will depend upon a variety of information such as the transform being used and the target application.

- Entropy coding is used to store the remaining data in an efficient way using Run Length Encoding (RLE) scheme or other Variable Length Coding (VLC) schemes such as Huffman or Arithmetic coding. The output of this stage is the final compressed image.

To decode the image, the above process is repeated in reverse, which then forms an approximation to the original image. Most image CODEC's work in a similar way to reduce the amount of data.

2.3.4 Digital Colour Representation

Most CODEC's follow the International Telecommunications Union (ITU) Recommendation 601 for representing colour in the digital domain [13]. That is instead of transmitting RGB colour information; the luminance (Y) and chrominance (Cr, Cb) are transmitted. This is known as YCrCb, but is often referred to as YUV. It should be noted though that this is similar but not identical to the analogue representation of YUV (see section 2.2.1). Luminance and chrominance colour spaces are used, as they are generally easier to compress than RGB. The following formula is used to obtain the luminance from the RGB components:

$$Y = \frac{77}{256}R + \frac{150}{256}G + \frac{29}{256}B \tag{2.7}$$

where R is the red component, G is the green component and B is the blue component. The luminance describes the intensity of the frame and is essentially a greyscale version of the original frame. As equation 2.7 shows, this is done by taking a portion of the red, green and blue frames to form the luminance frame. This equation has been derived through psychovisual testing performed by Commission Internationale de l'Eclairage (CIE). The colour information is called chrominance (Cb, Cr) is provided by two channels:

$$Cr = \frac{131}{256}R - \frac{110}{256}G - \frac{21}{256}B + 128 \tag{2.8}$$

$$Cb = -\frac{44}{256}R - \frac{87}{256}G + \frac{131}{256}B + 128 \tag{2.9}$$

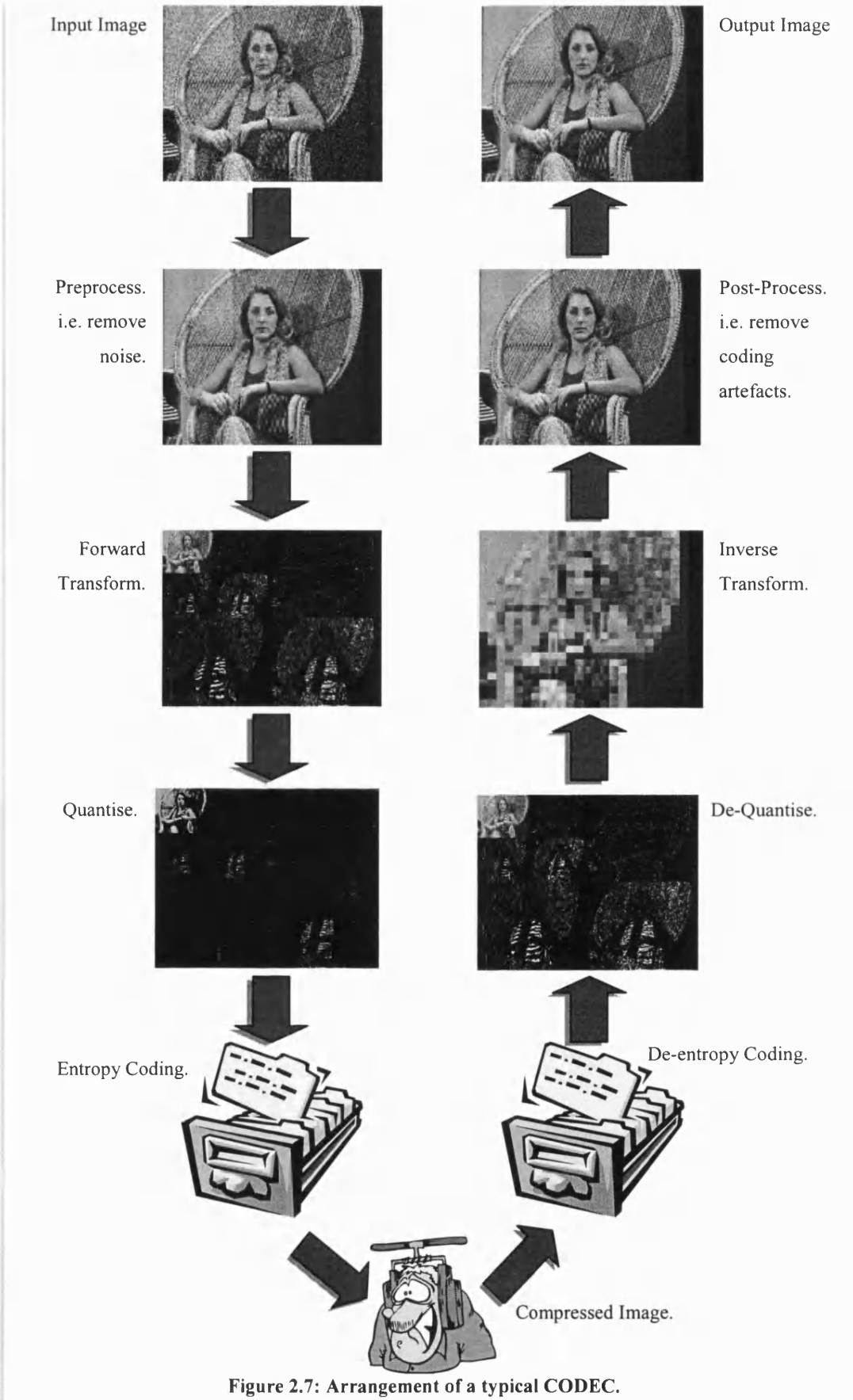


Figure 2.7: Arrangement of a typical CODEC.

Several other colour spaces, such as HIQ, HSV and YIQ can be used [6]. In addition, as in the PAL system, because the HVS is less sensitive to colour, the resolution of the chrominance channels can be reduced. This is done by subsampling, producing a format known as YCrCb 4:2:2. In its simplest form, this throws every other pixel in the horizontal plane away as shown in Figure 2.8. The whole process is shown in Figure 2.9. There are numerous other sub-sampling methods such as 4:2:0 and 4:1:1. It should be noted that there are other methods that can be used to compress and transform the image data [24].

2.3.5 Video Compression

As shown in Figure 2.1, a video sequence is made from a series of still images. Thus the most obvious method of compression is to use a still image CODEC such as the JPEG or JPEG 2000 CODEC. A frame compressed in such a way is usually called an Intra or I frame. Several methods exist that use this method of compressing frames such as Motion-JPEG (MJPEG). This method allows easy editing, for postproduction, of any frame and for this reason is used by many studios. This similarity between frames can be exploited by transmitting only the changes between frames, which is accomplished by using motion estimation. Figure 2.6 shows how similar adjacent frames are in a video sequence. The current frame is split into non-overlapping blocks as shown in Figure 2.10. Each block is examined and a vector assigned to it to map the block in the previous frame to it that is the closest map. Once obtained, the motion vectors are encoded and transmitted instead of compressing the whole frame as an I-Frame. This frame is called a Predicted-Frame (or P-Frame). However, the prediction rarely gets a perfect match and so there is an error between what should have been sent and what has been sent. This error is commonly known as a residual, which is then compressed using a still compression method and transmitted alongside the motion vectors. There is a problem here in as much as how much bandwidth should be assigned to motion vectors and how much to the residual.

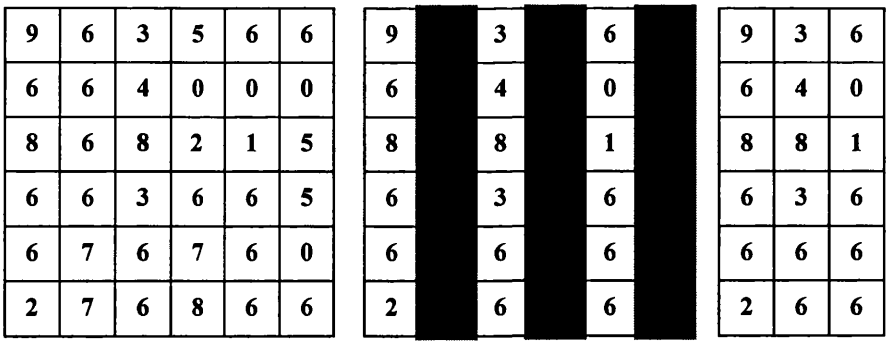


Figure 2.8: Basic sub-sampling.

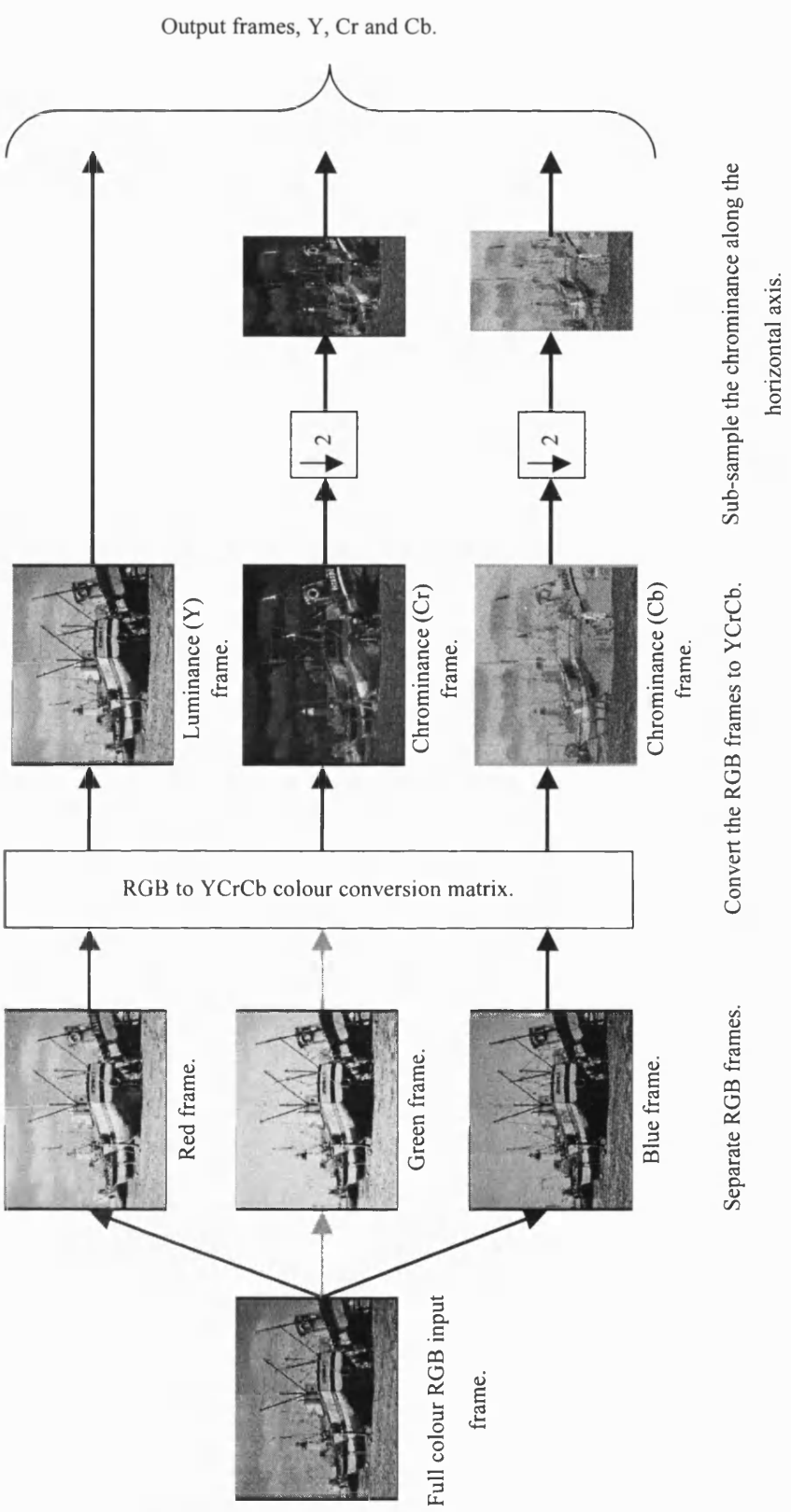


Figure 2.9: RGB to YCrCb 4:2:2 colour conversion process.

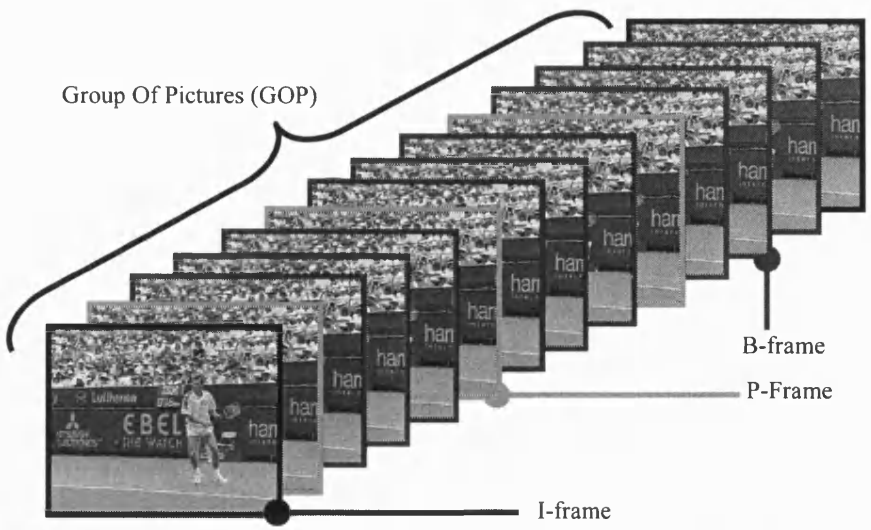


Figure 2.10: Example of how frames are grouped together in MPEG-II.

Most video compression systems use this technique, although many will have some slight variations. The following section describes the MPEG-II CODEC. There are several other commonly used CODEC's that work on a similar method such as MPEG-IV, H261, H263 and H26L **[25]**. In addition there are numerous standards and regulations that are used for digital video, especially when used for Digital Television **[26]**.

2.3.6 Moving Pictures Experts Group Video CODEC

One of the most common standards for digital video compression is the MPEG-II standard, which was established by the International Standards Organisation (ISO) [18]. All of the MPEG standards do not describe how to achieve each stage of the CODEC, but rather define how the data should be represented. For example it describes how motion vectors and bit-streams should be labelled. This allows for a large variety in implementations of the encoder. As long as the encoder generates a compliant bit-stream, any compliant decoder can then decode the stream.

The MPEG CODEC uses the DCT to compress images. Typically a compression ratio of 10:1 or 0.8 bits per pixel (bpp) is used for digital broadcasting. This works by creating a Group of Pictures (GOP) as shown in Figure 2.10, which in this case is 13 frames. The first frame is compressed using a still image compression scheme such as JPEG creating an I-Frame. The following frame is a P-Frame created using the motion compensation described earlier but with the constraint it can only be created from either another P Frame or an I Frame. The following three frames are created by using Bi-Directional Prediction. This allows a frame to be predicted by using both the previous P or I frame and the next P or I frame which creates a Bi-Directional Predicted Frame (or B-Frame). This is repeated two more times, that is another P frame is transmitted and then three B frames. The whole sequence is then started again with an I frame. Different systems use different methods to represent the remaining data. For example a 3D wavelet [27] could be used to represent several frames.

MPEG exploits the temporal redundancy by using motion compensation. The first way to do this is to use previous knowledge to predict the next frame, which is then labelled as a P-Frame. MPEG-II also uses past and future frames to predict a frame, which is labelled as a B-Frame. All three types of frames are used in MPEG-II to form a group of pictures as shown in Figure 2.10. This is a simplified view of how compression systems work. There are disadvantages to the method described:

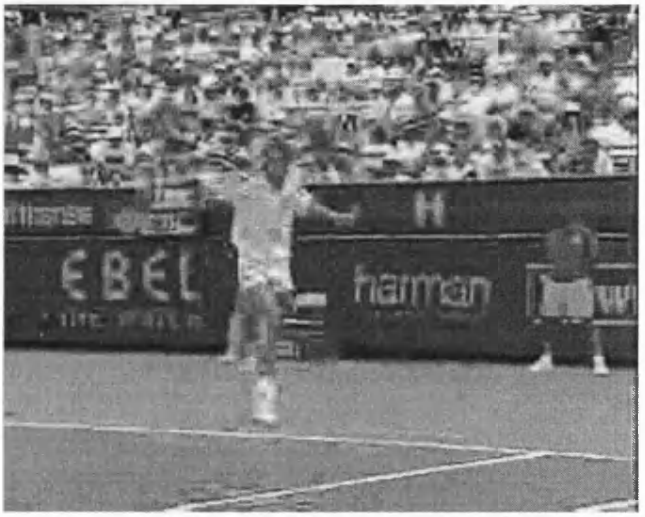
- If the bandwidth is too small, then there is a possibility that some important information may not be able to be transmitted. This can cause blockiness, ringing, dropout and various other artefacts as shown in Figure 2.11. Hence postprocessing is used to remove these distortions.

- Noise can cause the DCT to create more high frequency components than is necessary. This will reduce the efficiency. Hence preprocessing is used to eliminate as much noise and perceptually redundant information.

2.4 Conclusions

This chapter has introduced and shown the history of television. The differences between analogue and digital video systems has been demonstraighted and the need for compression in order to achieve the data rate necessary for transmission and storage of digital information has also been shown. The basic operations of still and video CODEC's have been discussed and the MPEG-II compression has been introduced. Some of the disadvantages of digital systems have been shown, such as the introduction of artefacts, which result in the degradation of the video. This highlights the need for pre and post processing systems in order to reduce or remove artefacts from digital video.

a) Uncompressed frame from the Stefan sequence.



b) Frame showing blocking as a result of MPEG-II compression.

c) Frame showing ringing and drop out artefact as a result of a wavelet CODEC.

Figure 2.11: Compression artefacts of frame 25 from the 352 x 288, YUV 4:2:0 (12bpp) Stefan sequence.

3 The Human Visual System

Knowledge of the HVS and the workings of the human brain can be exploited by an image processing system to improve performance whilst being visually lossless [6], [28] - [30]. For example, using a model of the HVS to design quantisation tables for JPEG, which results in better performance and less visible degradation when compared to the standard quantisation tables [31]. Other uses of modelling the HVS include watermarking and image mosaics [32], [33]. Despite their obvious differences, the HVS and image systems share a number of basic principles and operations [34]. This chapter describes the HVS, and how to measure its performance by using psychovisual evaluations.

3.1 Visual Perception

As stated earlier, the HVS is far from perfect. Simple drawings, illusions, can fool the HVS into seeing something that is not there as shown in Figure 3.1. The image on the left is normally viewed by the HVS to show a contour between the two gratings, despite the fact that there is no difference in the contrast. Similarly, the image on the right is normally viewed as four circles with a square in the middle, again there is no actual contour connecting these circles. It can be seen how easy it is to fool the HVS and how difficult it is to understand how it works. However, it is still unclear as to what is to blame for this, that is, is it the eye, the brain or both. However, some of the workings of the HVS are understood. For example it is known that the response to changes in intensity is non-linear. Consider a patch of light which has an intensity of $I + \Delta I$ with a background intensity of I (see Figure 3.2), the Just Noticeable Difference (JND), ΔI, can be determined as a function of I. The ratio $\Delta I / I$ is called the Webber ratio, which has a nearly constant value of 0.02. However, this does not hold for very high or very low intensities.

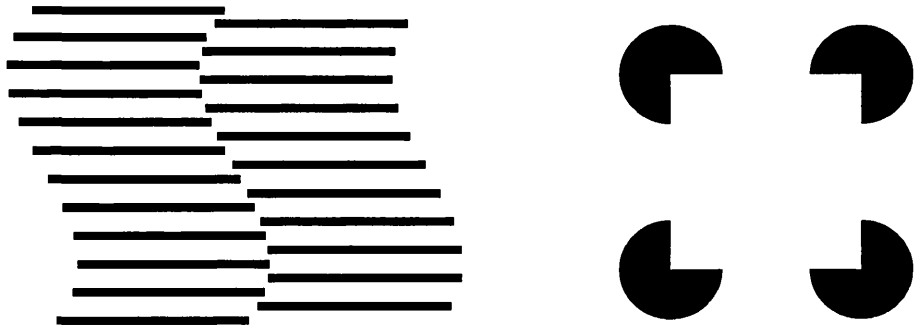


Figure 3.1: Demonstrations of visual illusions.

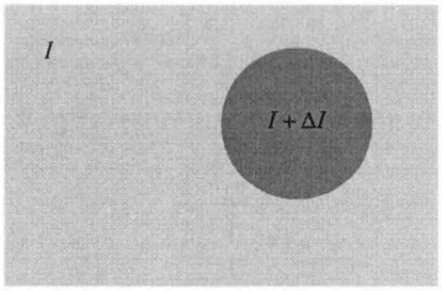


Figure 3.2: Example of the Webber ratio.

3.2 Visual Evaluation

Section 3.1 showed how the HVS works and the associated problems with how easily it can be fooled. In image processing, it is often a requirement to measure the quality of an image. There are two methods, *Objective* and *Subjective*, that can be used to measure the quality/performance of an image processing system. In most cases the original image, that is the image put into the image processing system, and the degraded image, that is the output of the image processing system, are used. These two images can then be used in either method to determine the performance of the system. Neither method requires knowledge of how the system works, and as such the processing system can be treated as a black box as far as evaluation is concerned.

3.2.1 Objective Evaluation

Objective evaluation is easy to implement and widely used today, although psychovisual evaluations are becoming more popular. Given that two images are available, described above, that is the input to the system and the output from the system, the simplest measure of performance is to take the absolute difference between the two images and then sum the result. This is known as the Sum of Absolute Differences (SAD) and is given by:

$$SAD(i,o) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left|i_{x,y} - o_{x,y}\right| \tag{3.1}$$

where i is the input image, o is the output image, M is the width in pixels of the image and N is the height of the image in pixels. Note that both images must have the same dimensions. The lower the result, the closer the two images are, with a value of 0 corresponding to identical images. Unless stated otherwise, all methods described here generate results in this way. This method is often averaged to give the Mean Absolute Difference (MAD), as given equation 3.2. There are however, three more commonly used measurements, similar to the MAD. The first is known as the Mean Square Error (MSE). This simply squares the differences to give a measure of power instead of using the absolute values and is calculated using equation 3.3.

$$MAD(i,o) = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left|i_{x,y} - o_{x,y}\right|}{NM} \tag{3.2}$$

where i is the input image, o is the output image, M is the width in pixels of the image and N is the height of the image in pixels.

$$MSE(i,o) = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left(i_{x,y} - o_{x,y}\right)^2}{NM} \tag{3.3}$$

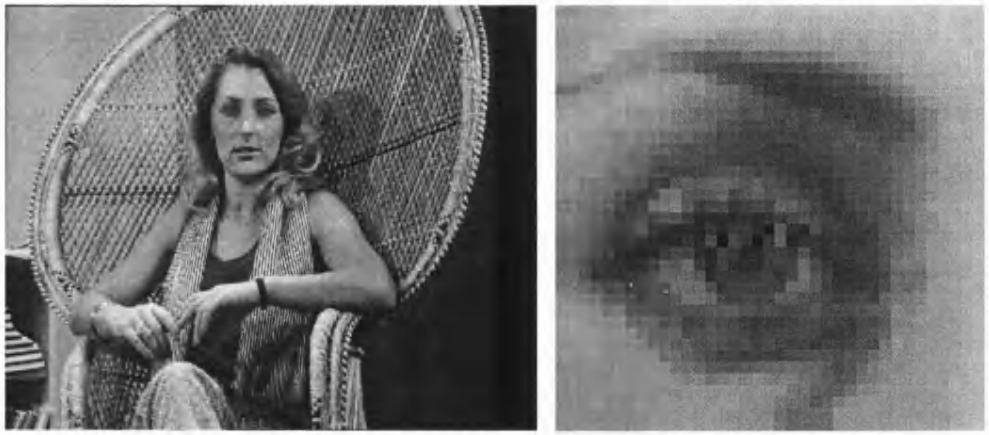
where i is the input image, o is the output image, M is the width in pixels of the image and N is the height of the image in pixels. The second method, called Root Mean Square Error (RMSE) is more commonly used than the MSE. This is simply the square root of the MSE as given by:

$$RMSE(i,o) = \sqrt{MSE} = \sqrt{\frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left(i_{x,y} - o_{x,y}\right)^2}{NM}} \tag{3.4}$$

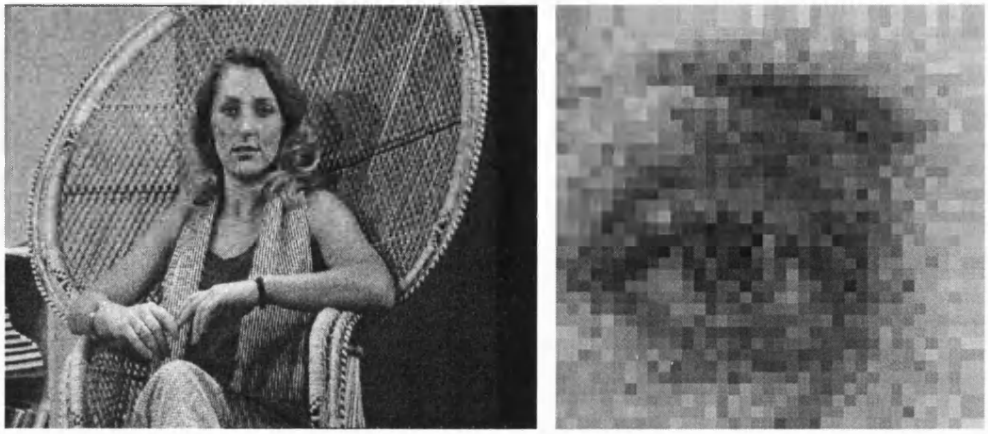
where i is the input image, o is the output image, M is the width in pixels of the image and N is the height of the image in pixels. This is a useful method as it gives an average value of the pixel-by-pixel difference between the input (i) and output (o) images. The third method is known as the Peak Signal to Noise Ratio (PSNR). This calculates the ratio, in decibels, of the peak signal power to the noise power and is given by:

$$PSNR(i,o) = 10\log_{10}\left(\frac{\max(value)^2}{MSE(i,o)}\right) = 20\log_{10}\left(\frac{\max(value)}{RMSE(i,o)}\right) \tag{3.5}$$

where i is the input image, o is the output image and $\max(value)$ is the maximum possible amplitude of the image, which is 255 for 8bit images. Unlike previous methods, the higher the result, the more similar the images. This is better than the straight forward Signal to Noise Ratio (SNR) measure as the average intensity does not affect the result. There is no restriction on which measurement is used for any given application as the MSE, RMSE and PSNR all measure exactly the same measurements, although on different scales. The Barbara 2 test image is used (see Figure 3.3), corrupted with Gaussian noise (see section 5.1.1), to show examples of these measurements (see Table 3.1).



a) Input image – Barbara 2 at a resolution of 720 x 576 pixels and 8bit greyscale.



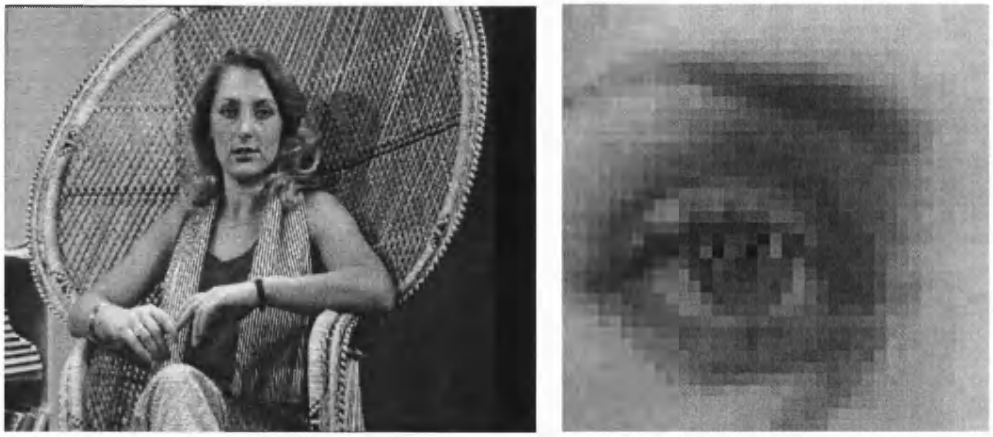
b) Image corrupted with Gaussian noise.

Figure 3.3: The use of objective measurements.

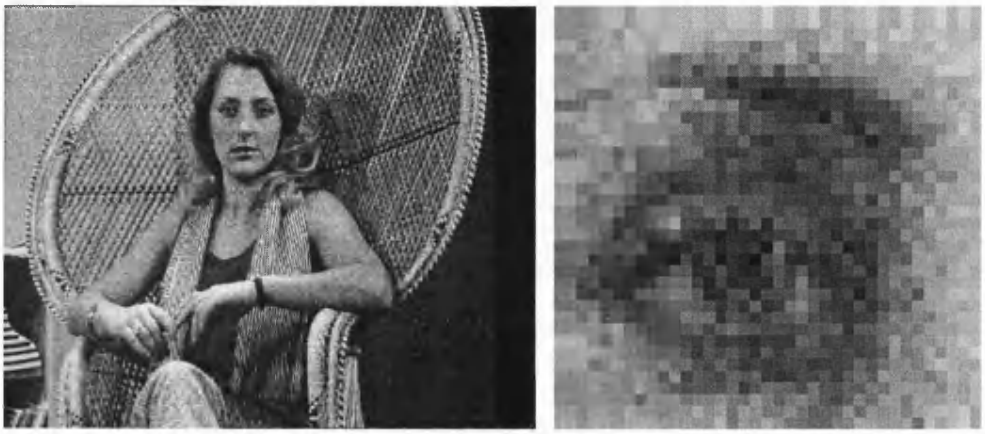
Measurement used	Result
SAD	3295518.00
MAD	7.95
MSE	99.95
RMSE	10.00
PSNR	28.13dB

Table 3.1: The use of objective measurements.

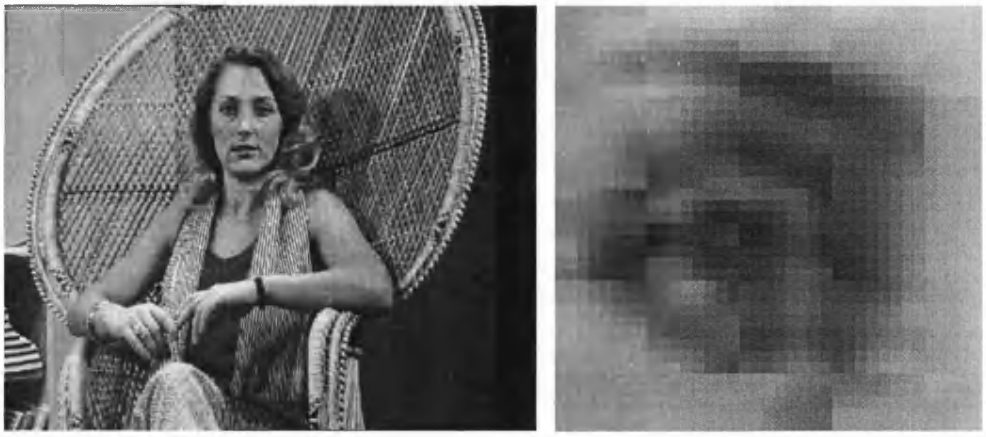
Objective measurements are the easiest and fastest method to use to evaluate the performance of a system. These measurements are easily repeatable and the results will not change given the same pair of images. However, in some instances, the objective measurements do not reflect the true performance. For example, Figure 3.4 shows the test image corrupted by Gaussian noise and also corrupted by the JPEG compression. Using the MSE, objective measurements indicate that the JPEG image is better than the image corrupted with noise. However, when observers are asked to look at these two images, they say that the noisy image is more acceptable than the JPEG. Brainard discusses some of the properties of the visibility of noise further in [35].



a) Input image – Barbara 2 at a resolution of 720 x 576 pixels and 8bit greyscale.



b) Image corrupted by Gaussian noise with an MSE of 99.95.



c) JPEG compressed image with an MSE of 78.87.

Figure 3.4: Example of the problems with objective measures, which say that JPEG (c) is better than the noisy image(b). However, psychovisually, the noisy image (b) is often preferred over the JPEG (c).

3.2.2 Subjective Evaluation

Subjective evaluation provides a method of obtaining the true performance of a system. As all subjective evaluation methods require a user to assess the quality of the test images, these methods are commonly referred to as **'Psychovisual Evaluations'**. This area has been the subject of much research and has been applied to several applications such as television [30], [36] - [39], [41]. The basic principle is to get an assessor(s) to view an image(s) and then vote on the quality of the image(s). The voting, or feedback is then given on how the assessors rate the performance of the system. This is considered to be a better method of evaluating the performance of a system as compared to objective evaluation. However, these test often take a considerable amount of time to set-up, run, and analyse. There are two basic types of subjective evaluation:

- Assessments to establish the performance of systems under optimum conditions.
- Assessments of the ability of a system to retain quality under non-optimum conditions.

There are however, several ways in which each test can be performed [36], [37]. The following chapters give a brief overview of the methods for implementing and settings for visual tests to be carried out in.

3.2.2.1 Double Stimulus, Impairment Scale

The Double Stimulus, Impairment Scale (DSIS) method uses test sets of pairs of images. Each pair of images consists of an un-impaired (or more commonly known as the reference) image and an impaired image. Test sets are then created for every condition that the system is to be tested under. For example, if a new image CODEC is to be tested, then there may be several test sets, one for a varying a quantiser table, one for the compression ratio used, etc.

Before testing begins, a training sequence should be shown. This simply shows a series of test images that cover the range of impairments. These are then shown in pairs (unimpaired and impaired) to show the assessors the range of quality of the images. The test images used for this purpose should not be used in the actual test itself. At the start of a test session, about 5 dummy pairs are shown. This allows the assessors' responses to settle. The results of these 5 test pairs should not be used when analysing the results. In this method, a session should last up to 30 minutes with at least 15 people taking part. Testing is actually a relatively simple procedure, as illustrated in Figure 3.5, and described below:

1. Select a test pair at random,
2. Show the reference image for 10 seconds,
3. Show an Inter Stimulus Interval (ISI) for 3 seconds,
4. Show the impaired image for 10 seconds,
5. Finally allow the assessors to vote for 5 to 11 seconds.

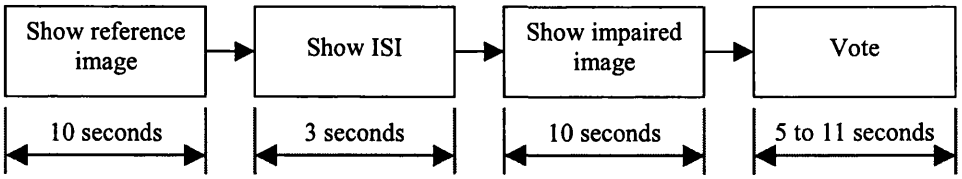


Figure 3.5: Stages of visual testing using the DSIS method.

There is a variant of this method that simply shows the test pair twice, as indicated by the dotted line in Figure 3.5. Voting is made by comparing the impaired image against the reference image using the scale shown in Table 3.2. This scale is known as the five-point scale. The ISI is defined by as screen filled completely with a mid-grey level. This method of testing is used when a full range of impairments is available from the system under test.

Score	Description
5	Imperceptible
4	Perceptible, but not annoying
3	Slightly annoying
2	Annoying
1	Very annoying

Table 3.2: Five-point scoring scale.

3.2.2.2 Double Stimulus, Continuous Quality Scale

The Double Stimulus, Continuous Quality Scale (DSCQS) is used when it is not possible to get test images that cover the full range of quality of the system under test. Reference and impaired images are mixed, that is that the first image to be shown (image *A*) can be the reference or impaired with the second image (image *B*) being the opposite. This is done at random and it must be kept track of which is the reference. Again, the order of the images is random and also the training and dummy images are used before the test starts. Two ways exist in which to implement this method:

- The assessor is allowed to switch between the two images (*A* and *B*) until they are satisfied that they have formed a suitable opinion of each. This allows for only one assessor at a time.

- A test pair is shown twice as in DSIS. This allows for several assessors to perform the test in one sitting. Unlike DSIS, the assessors are allowed to vote any time during the second viewing.

The timing is the same as for DSIS, however, the scoring is slightly different. Each assessor is asked to put a mark on a pair of vertical scales to rate the quality of each image in the test pair as shown in Figure 3.6. This method of scoring allows a continuous quality grading and helps to avoid quantising errors. The scale is split into five sections corresponding to the five-point scale and are included for

guidance only. In analysing these results, the scoring is changed to a number in the range of 0 to 100. The difference between the reference and impaired is then used. There are several methods that may be used to analyse the results [36], one of which is shown in section 3.2.2.4.

3.2.2.3 Other Methods

Other testing and scoring methods exist, for example there are two methods available for testing single images, Single Stimulus (SS) and Single Stimulus, Multiple Repetition (SSMR). One of these other scoring methods is known as the comparison scale. In this method, the assessor simply grades the second image (B) with respect to the first image (*A*). The scoring is fairly simple as shown in Table 3.3. Essentially, this method simply indicates if the assessor saw a difference between the two images, and if so, how much better/worse it is.

Score	Description
-3	Much worse
-2	Worse
-1	Slightly worse
0	The same
1	Slightly better
2	Better
3	Much better

Table 3.3: The comparison scale.

3.2.2.4 Analysing the Results

There are several ways in which results from a visual test may be analysed. The most common method, as used by the ITU, is to use the Mean Opinion Score (MOS) and the 95% confidence interval [36]. Considering that each test session has several test conditions (*j*), for example varying bit rate, each test condition may be applied to several different images (*k*) and that these may be repeated several times (*r*), then the MOS is defined as given in equation 3.6.

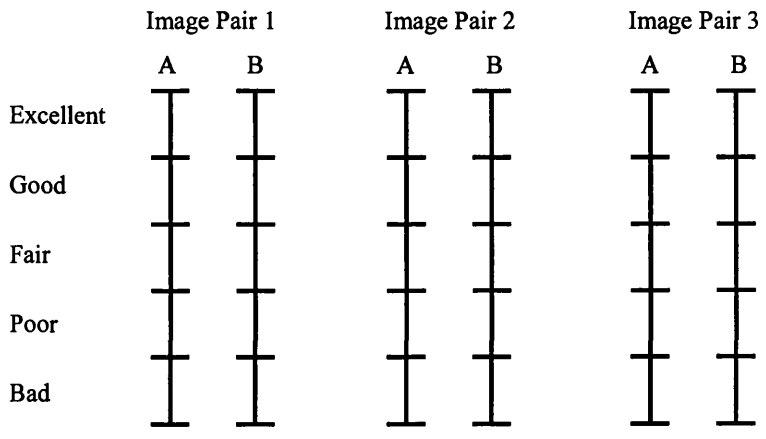


Figure 3.6: The scoring scale used in DSCQS.

$$\bar{u}_{jkr} = \frac{\sum_{i=1}^{N} u_{ijkr}}{N} \tag{3.6}$$

where u_{ijkr} is the score of assessor i for test condition j using image k on repetition r, and N is the number of assessors. In conjunction with this, the 95% confidence interval is used as given by:

$$\left[\bar{u}_{jkr} - \delta_{jkr}, \bar{u}_{jkr} + \delta_{jkr}\right] \tag{3.7}$$

where:

$$\delta_{jkr} = 1.96\frac{\sigma_{jkr}}{\sqrt{N}} \tag{3.8}$$

and the standard deviation (σ), is given by:

$$\sigma_{jkr} = \sqrt{\frac{\sum_{i=1}^{N}\left(u_{ijkr} - \bar{u}_{jkr}\right)^2}{N-1}} \tag{3.9}$$

3.3 Conclusion

This chapter has shown the basic principles behind the HVS and how to measure the performance of an image processing system, both objectively and subjectively. Objective measures are used due to their speed and offer a direct comparison between results. Subjective measures are used to obtain a more accurate result of what an observer will see. However subjective measures often take a long time to obtain and can be difficult to compare against other subjective results. The noise removal stages (see section 7.1 and chapter 9) of this project will use some form of objective measure, simply for the speed in development it offers. Subjective evaluation will be used to determine a psychovisually lossless preprocessing system (see section 7.2) whilst still filtering the image as much as possible.

4 Mathematical Morphology

Mathematical Morphology is the analysis of signals in terms of shape. This simply means that morphology works by changing the shape of objects contained within the signal. Mathematical morphology was developed in the 1970's [42] by *G.Matheron* [43] and *J.Serra* [44] and has several advantages over other techniques especially when applied to image processing as outlined below:

- Works by using shape-based processing,
- Can be designed to be idempotent,
- Computationally efficient

Mathematical morphology has widely been used in image enhancement and restoration [45] - [50]. Harvey and Marshall use morphological filters for removing scratches and other degradations from archived film material [46], [50]. Image segmentation is an application well suited to morphological operators [51] - [53]. Image simplification and noise reduction have also been demonstraighted by the use of morphological operators [54]. Both of these methods could be used as simple preprocessing steps for a CODEC. Mathematical morphology has also been used to attempt to form a CODEC with varying degrees of success [55], [56]. Several other applications have also proven the usefulness of mathematical morphology such as, texture analysis [57], template matching [58], range imagery [59] and even fractal landscape generation [60]. There are many more applications that morphology can be applied to [61]. Mathematical morphology has been widely researched for use in image and video processing, but with no application to preprocessing for increasing a CODEC's performance. The rest of this chapter covers the basic principles behind Morphology, building up from simple 1D binary functions to 3D grey-scale functions.

4.1 Binary Morphology

4.1.1 Dilation

Morphology uses Set Theory as the foundation for many functions. The simplest functions to implement are Dilation and Erosion [62]. Dilation in 1D is defined as:

$$A \oplus B = \left\{ x : (\hat{B})_x \cap A \neq \varnothing \right\} = \bigcup_{x \in B} A_x \tag{4.1}$$

where A and B are sets in $\mathbb{Z}$. Equation 4.1 is also known as Minkowski Addition and simply means that B is moved over A and the intersection of B reflected and translated with A is found. Usually A will be the signal or image being operated on and B will be the Structuring Element (SE). Equation

4.1 is used to process binary sets of data. Dilation has several interesting properties, which make it useful for image processing. These properties are:

- Translation invariant.

  This means that the result of A dilated with B translated is the same as A translated dilated with B as given by:

$$(A \oplus B)_x = A_x \oplus B \tag{4.2}$$

- Order invariant / Associative.

  This simply means that if several dilations are to be done, then the order in which they are done is irrelevant. The result will be the same irrespective:

$$(A \oplus B) \oplus C = A \oplus (B \oplus C) \tag{4.3}$$

- Increasing operator.

  This means that if a set, A, is a subset of another set, B, then the dilation of A by C is still a subset of B dilated by C:

$$A \subseteq B \Rightarrow A \oplus C \subseteq B \oplus C \tag{4.4}$$

- Scale invariant.

  This means that the input and structuring element can be scaled, then dilated and will give the same as scaling the dilated output:

$$rA \oplus rB = r(A \oplus B) \tag{4.5}$$

  where r is a scale factor.

- Commutative:

  This implies that the order of the operation is irrelavent:

$$A \oplus B = B \oplus A \tag{4.6}$$

These properties can be very useful in image processing and can result in some operations being simplified. Figure 4.1 shows how dilation works on a 1D binary signal. The structuring element shown in Figure 4.1 uses the value of the elements immediately to the right and left of the current element (the structuring element in this case looks for ones on the input sequence). Any shape or size-structuring element can be used, where an element with the value of 0 indicates that the corresponding element in set A is not to be used, and a value of 1 indicates that it is to be used. For example, the structuring element shown could be considered to have 0's on the extreme left and right, as the

corresponding inputs would be ignored. The output is given by equation 4.1 and will be set to one unless the input is the inverse of the structuring element. For example, '000' would cause the output to be zero. The output is placed at the origin of the structuring element as shown.

From Figure 4.1, it can be seen that dilation will completely remove any runs of zeros less than the length of the structuring element (this is only for this type of structuring element though). Longer runs of zeros are shortened at their extremities. Note there is a problem with this technique. That is, *what happens at the borders*? In general, the structuring element is offset so that it is completely contained within the signal. However, this results in the output containing less data than the input, which is not desirable for image processing where the whole image is to be modified. Although there are several ways of overcoming this problem, the chosen method is to detect when the structuring element is at the edge, and only use information from the parts of the structuring element that still lay within the signal.

4.1.2 Erosion

The opposite of dilation is known as erosion and is defined as:

$$A \ominus B = \{x : (B)_x \subseteq A\} = \bigcap_{x \in B} A_x \tag{4.7}$$

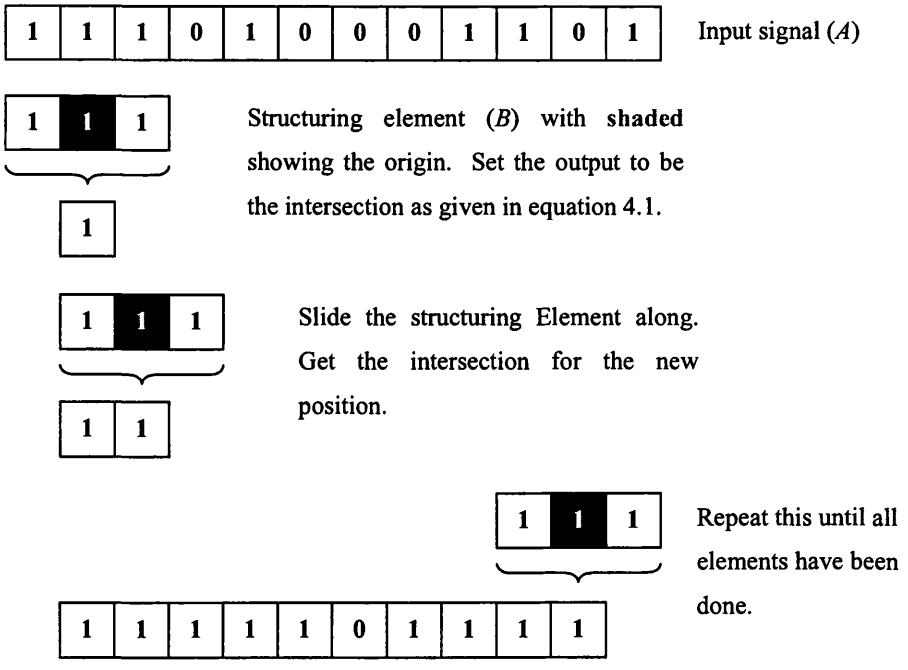


Figure 4.1: An example of the Dilation operation.

This definition is also known as Minkowski Subtraction. The equation simply says, erosion of A by B is the set of points x such that B translated by x is contained in A. Figure 4.2 shows how erosion works on a 1D binary signal. This works in exactly the same way as dilation. However, equation 4.7 essentially says that for the output to be a one, all of the inputs must be the same as the structuring element. Thus, erosion will remove runs of ones that are shorter than the structuring element. Erosion, like dilation also contains properties that are useful for image processing:

- Translation invariant.

  This means that the result of A eroded with B translated is the same as A translated eroded with B as given by:

  $$(A \ominus B)_x = A_x \ominus B \tag{4.8}$$

- Increasing operator.

  This means that if a set, A, is a subset of another set, B, then the erosion of A by C is still a subset of B eroded by C:

  $$A \subseteq B \Rightarrow A \ominus C \subseteq B \ominus C \tag{4.9}$$

- Scale invariant.

  This means that the input and structuring element can be scaled, then eroded and will give the same as scaling the dilated output:

  $$rA \ominus rB = r(A \ominus B) \tag{4.10}$$

  where r is a scale factor.

- Non-commutative:

  This is the opposite of the dilation in that the order of the erosion does matter:

  $$A \ominus B \neq B \ominus A \tag{4.11}$$

- Not-associative:

  This means that the order of multiple erosions matters:

  $$A \ominus (B \ominus C) \neq (A \ominus B) \ominus C \tag{4.12}$$

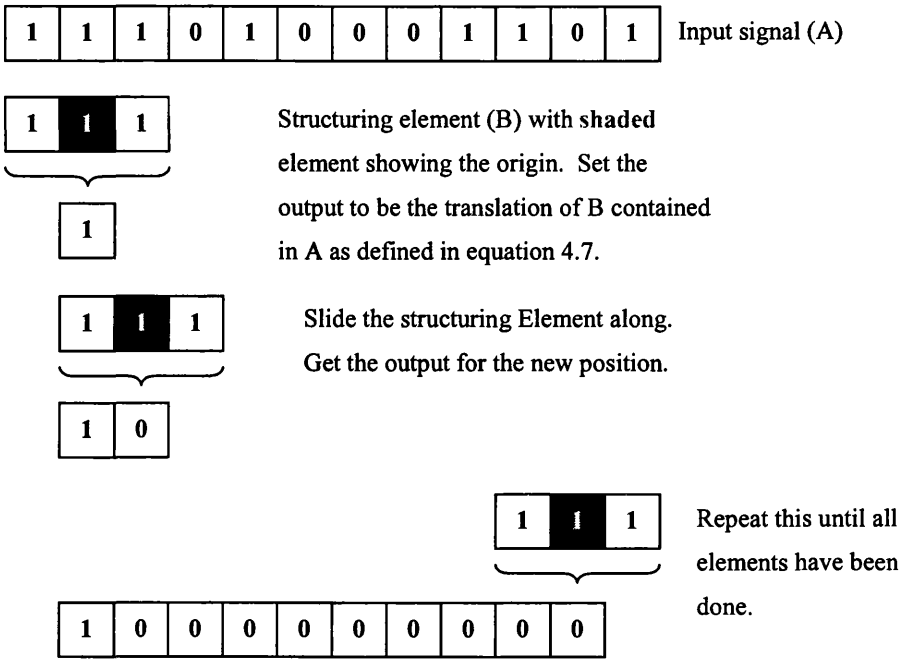


Figure 4.2: Example of how erosion works.

- Decreasing Operator.

  For a set A and two structuring elements, B_1 and B_2, the erosion of one as a subset of the other:

$$B_1 \subset B_2 \Rightarrow A \ominus B_2 \subset A \ominus B_1 \tag{4.13}$$

4.1.3 Opening and Closing

Both dilation and erosion have intresting and useful properties. However, it would be useful to have the properties of both in one function. This can be done in two ways. The first method , Opening [63], is defined by equation 4.14. This simply erodes the signal and then dilates the result as shown in Figure 4.3. As can be seen, the zeros are opened up and any ones that are shorter than the structuring element are removed with the rest of the signal left unchanged. This is a very useful property as it means that if the filter is applied once, no more changes to the signal will result from repeated applications. This property is called '**Idempotency**' and is given in equation 4.15.

$$A \circ B = (A \ominus B) \oplus B \tag{4.14}$$

$$(A \circ B) \circ B = A \circ B \tag{4.15}$$

The opposite of an opening, is a closing defined by equation 4.16. Figure 4.4 shows how this works from which it can be seen that this closes gaps in the signal in the same way as opening opened up gaps. Closing also has the property of being idempotent. Both of these filters again have interesting properties that would be nice to have in one filter. The opening and closing can be combined to merge these properties. There are two ways of combining these, the first of which is known as an Open-Close filter (see equation 4.17), which consists of first opening the signal and then closing the signal. The opposite can also be done by closing and then opening. This is called a Close-Open filter (see equation 4.18). Figure 4.5 shows how these filters work. It can clearly be seen that both filters remove any runs of ones or zeros that are shorter than the structuring element. The operations, Open-Close and Close-Open are both idempotent.

$$A \bullet B = (A \oplus B) \ominus B \tag{4.16}$$

$$A \odot B = (A \circ B) \bullet B \tag{4.17}$$

$$A \circledcirc B = (A \bullet B) \circ B \tag{4.18}$$

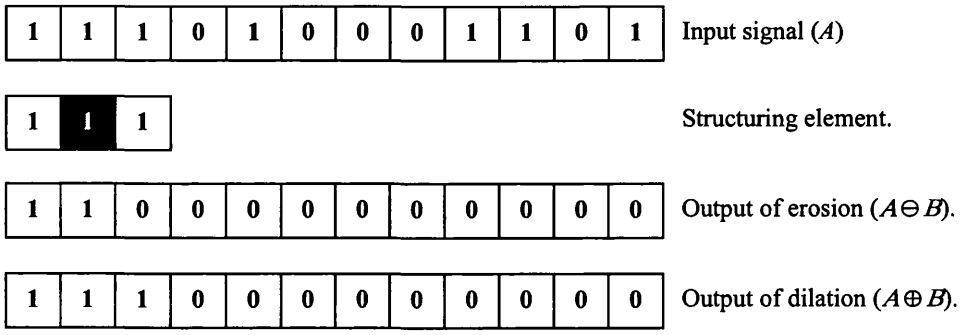


Figure 4.3: The opening operator.

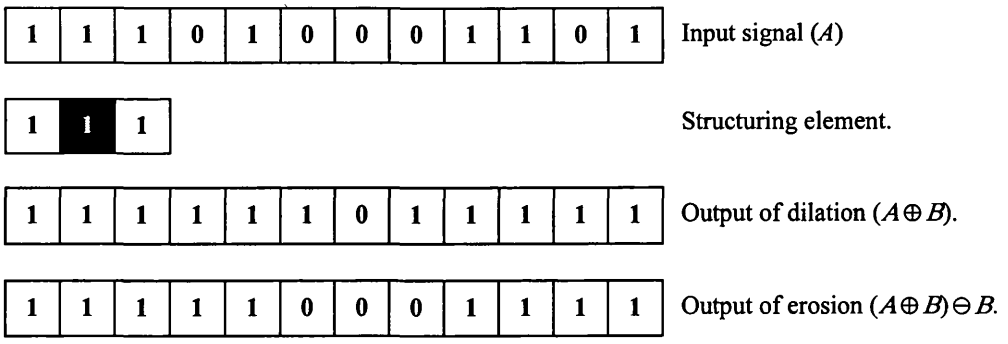


Figure 4.4: The closing operator.

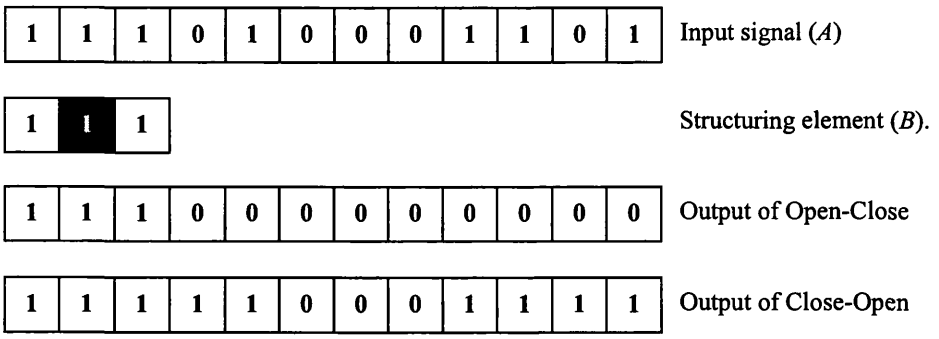


Figure 4.5: The Open-Close and Close-Open operators.

4.2 Greyscale Morphology

For morphology to be of use in image processing, it needs to be extended to non-binary signals. There are various ways in which this can be done [49]. The chosen method uses very simple functions, which allow them to be implemented in an efficient way. The following sections describe one method of implementing greyscale morphology. For comparison, the second most common method is briefly described here. It is often reffered to as level sets or threshold decomposition [64]. All greyscale signals and images are made from a number of discrete values (K). For example, an 8-bit greyscale image has 256 possible levels. A simple method of applying greyscale method is to simply convert the greyscale image to binary. Threshold decomposition takes a greyscale signal and converts each level into binary as given by:

$$T_h(f) = (x \mid f(x) \geq h) \tag{4.19}$$

where f is the greyscale signal and h is the level, $h = 0,1,2,...,K-1$. This results in each level being represented as a binary image or level set. Each level set can then be processed by using the binary morphological operator. Once each level set has been processed, they are combined using equation 4.20.

$$\hat{f}(x) = \max(h \mid x \in T_h(f)) \tag{4.20}$$

4.2.1 Greyscale Dilation

From equation 4.1 and Figure 4.1 it should be clear that this is actually taking the maximum value lying within the structuring element. Hence, dilation can be redefined for greyscale as given in equation 4.21. This works in exactly the same way as before, but just takes the maximum value laying within the 1's of the structuring element as shown in Figure 4.6. This method of using the values where the structuring element is 1, is known as a Flat Structuring Element (FSE). There is another

method [47] that is used in greyscale image processing. Unless stated otherwise, the method described above is used in this thesis, the following description is included as a reference only.

$$A \oplus B = \max_{i \in B, \forall x} (A_{x+i}) \tag{4.21}$$

As well as having the structuring element to choose the input elements to use, another set is used to allow a value to be added to the values used. This is sometimes written as one set as shown in Figure 4.7. The equation for this method is defined as:

$$A \oplus B = \max_{i \in B, \forall x} (A_{x+i} + B_i) \tag{4.22}$$

4.2.2 Greyscale Erosion

Like dilation, from equation 4.7 and Figure 4.2 it should be clear that all erosion is actually doing is taking the minimum value from within the structuring element. Thus, erosion can be redefined as:

$$A \ominus B = \min_{i \in B, \forall x} (A_{x+i}) \tag{4.23}$$

This can still be applied to binary signals, but more importantly, they can be applied to real numbers. The rest of the functions, opening, closing and so on remain unchanged, but use the above definitions (equations 4.22 and 4.23) for greyscale.

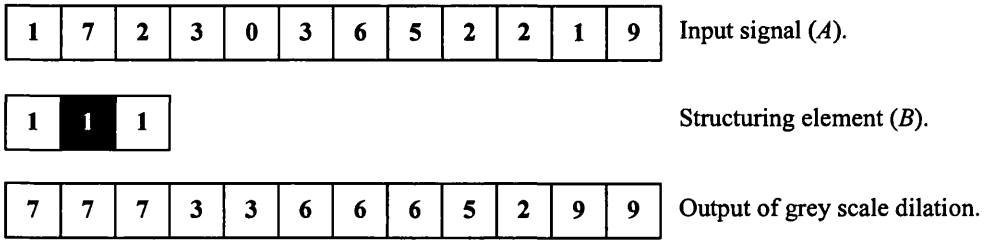


Figure 4.6: An example of greyscale dilation.

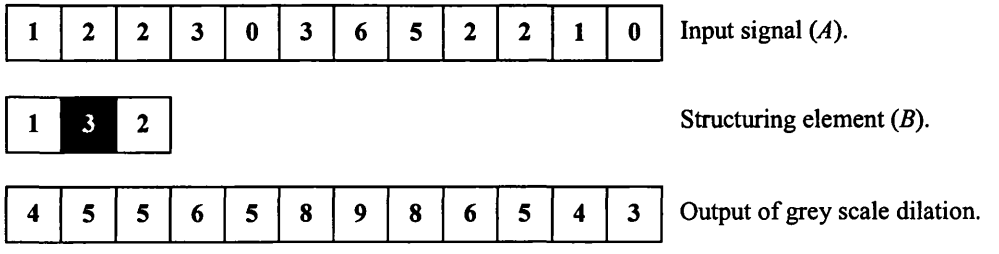


Figure 4.7: A variation on the standard dilation using a non-flat structuring element.

4.3 Two Dimensional Morphology

Now to be able to use morphology in image processing, the definitions need to be applied in two dimensions (2D). This can be done relatively easily as described below. The signal has now become a 2D signal, now called an image and hence the structuring element is changed to become 2D as shown in Figure 4.8 where the first structuring element shown is known as the *Four Nearest Neighbours* (4nn). This limits the operations to work only on horizontal and vertical components. To overcome this problem and allow diagonal components to also be used, the *Eight Nearest Neighbours* (8nn) can be used.

4.3.1 2D Dilation

Using the 2D SE, dilation can be redefined as given in equation 4.26. This still works in the same way as before as shown in Figure 4.9. The structuring element is moved across the image as before and the maximum value inside the structuring element is then set as the output (see Figure 4.10b).

$$A \oplus B = \max_{(i,j)\in B} \left(A_{x+i,y+j} \right) \tag{4.24}$$

4.3.2 2D Erosion

Erosion can also be redefined in exactly the same way, as dilation was to give a new 2D erosion definition as given by equation 4.25. This works in the same way as dilation with the exception of erosion takes the minimum instead of the maximum. Again, the other definitions of opening, closing, etc. remain unchanged. To use them in 2D, they must use the above methods for erosion and dilation. An example image processed using this is shown in Figure 4.10c.

$$A \ominus B = \min_{(i,j)\in B} \left(A_{x+i,y+j} \right) \tag{4.25}$$

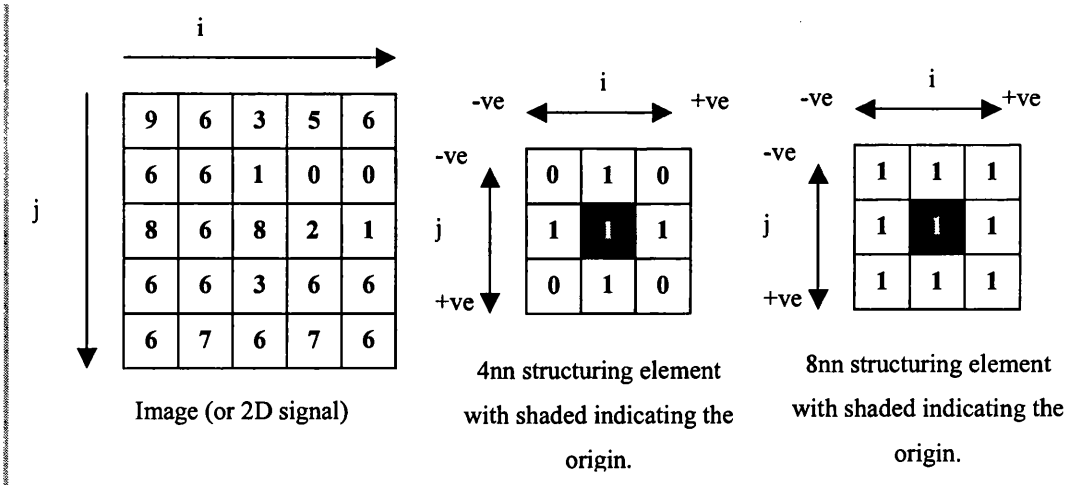


Image (or 2D signal)

4nn structuring element with shaded indicating the origin.

8nn structuring element with shaded indicating the origin.

Figure 4.8: 2D extension of the structuring element.

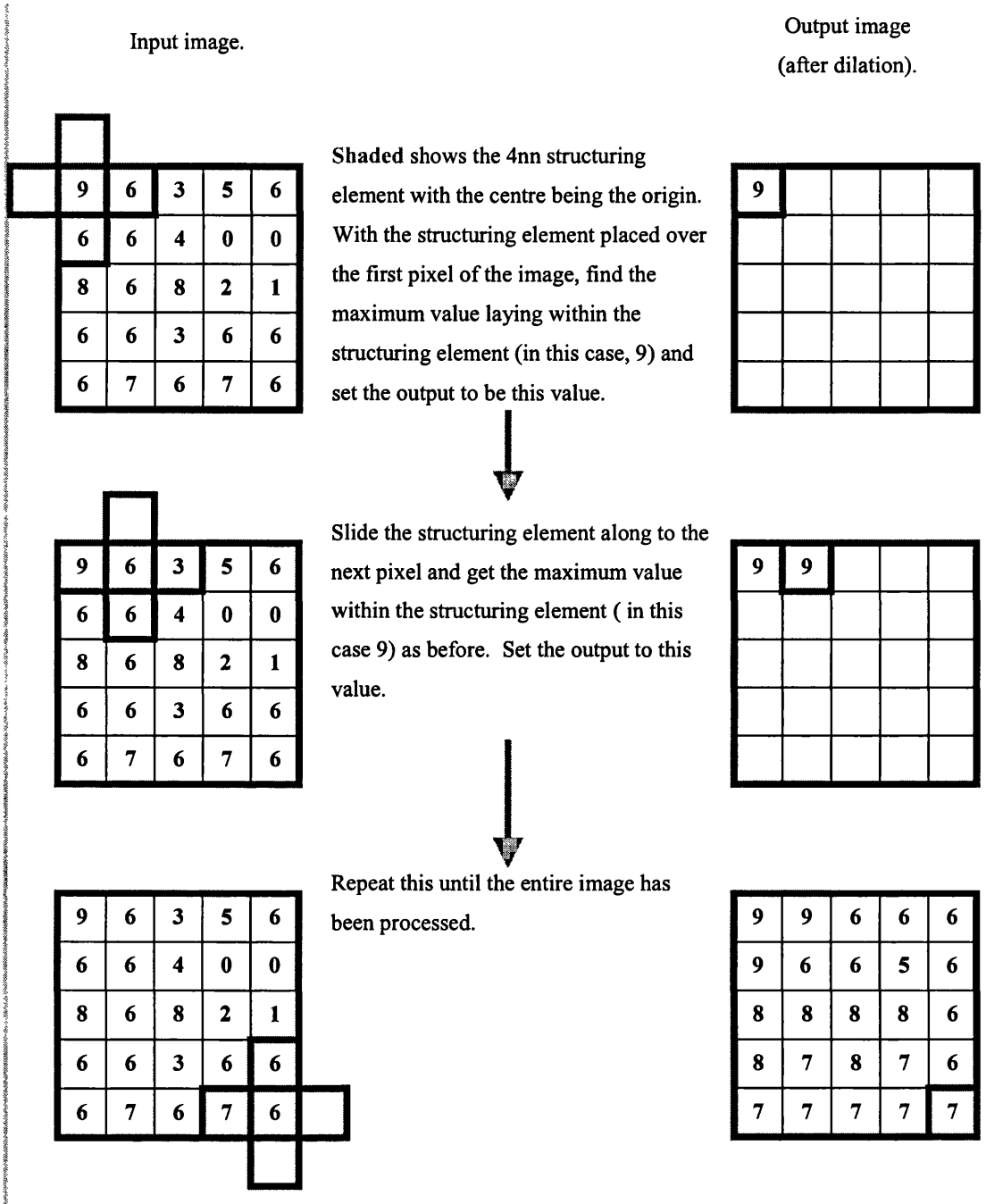


Figure 4.9: An example of 2D dilation.

4.3.3 Application to Image Filtering

These filters can now be applied to images as shown in Figures 4.10 – 4.12. It can be seen from these images exactly what the operations do. Dilation extends the boundary of object by removing low valued regions. Erosion contracts the boundaries by removing high valued regions. Distortions can bee seen in some of these images. The distortion generated not only depends on the filter but also on the image. Opening will remove high intensity points whilst keeping the rest of the image intact. Closing is doing the opposite of this, removing low valued points whilst keeping the rest of the image intact. Both Open-Close and Close-Open remove both high and low valued points while keeping the rest of the image intact. However, the last two operations do not give the same result due to the order of the erosions and dilations.

4.4 Three Dimensional Morphology

For video processing, it is beneficial to filter spatio-temporally (i.e. in Three Dimensions, 3D). This is done because noise is uncorrelated which means that if a pixel in the current frame is corrupted by noise, then there is a strong chance that the pixel will be of a different intensity value in other frames. Hence, by using the other frames to process the current frame, the noise can be filtered out. Thestructuring element is again extended to encompass an extra dimension as shown in . The 4nn and 8nn are extended to produce the 6nn and 26nnstructuring element. The 6nn is also referred to as 4_{nn}^{3D} and the 26nn is referred to as 8_{nn}^{3D}. This makes it easier to see the relationship between the 2D and 3D structuring element. Data in 3D can be thought of in two ways. The first is simply to think of 3D data as a real world object. For example, an object that has the three dimensions, width, height and depth. Most medical applications produce 3D, or volumetric, data such as the Magnetic Resonance Imaging (MRI) which often use mathematical morphology operators including the watershed to segment the data [65] - [67]. The second method is to think of 3D as a sequence of 2D images such as a video sequence. The 3rd dimension is then time and not depth.

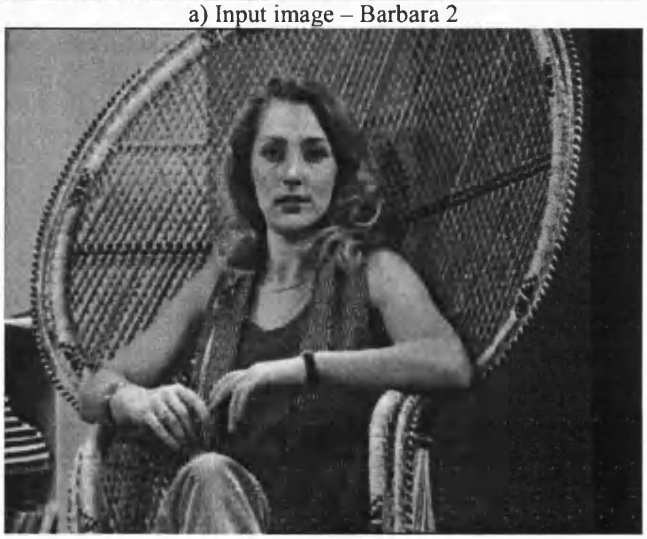
4.4.1 3D Dilation

Dilation can be easily extended to 3D signals, in a similar manner to how it was extended from 1D signals to 2D signals, by defining:

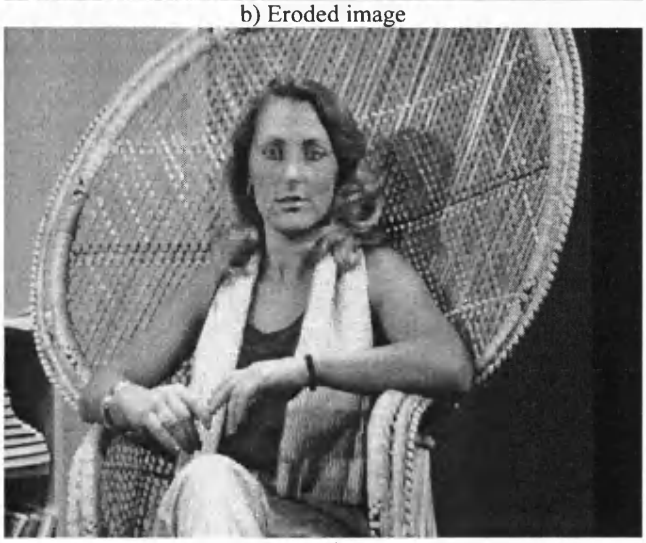
$$A \oplus B = \max_{(i,j,k) \in B} \left(A_{x+i,y+j,z+k} \right) \tag{4.26}$$

Since the image is 3D (i.e. a sequence), consiquently the structuring element is extended to 3D as shown in Figures 4.13 and 4.14. This is applied in the same way as before. The structuring element is placed on the first pixel of the first frame, the output is set to the maximum value lying within the structuring element and the structuring element is then moved across the current frame until the entire frame has been processed. Then the next frame is done in exactly the same way. The process is repeated until all the frames have been processed.

a) Input image – Barbara 2



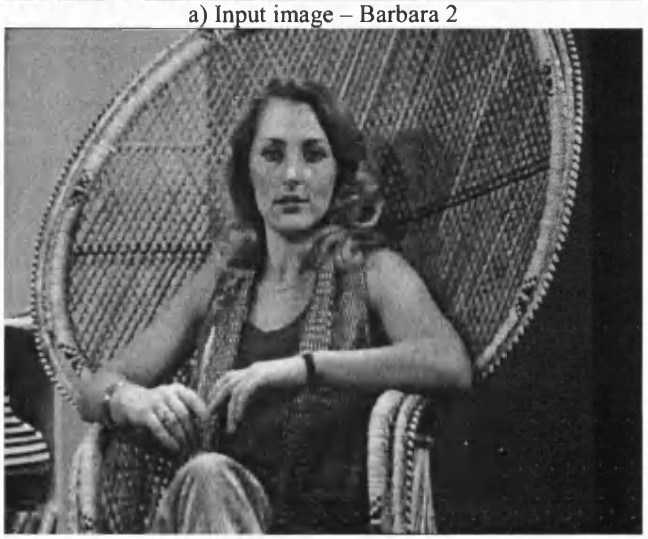
b) Eroded image



c) Dilated image

Figure 4.10: Basic morphological operations, dilation and erosion applied to the 720 x 576, 8bit greyscale Barbara 2 image.

a) Input image – Barbara 2

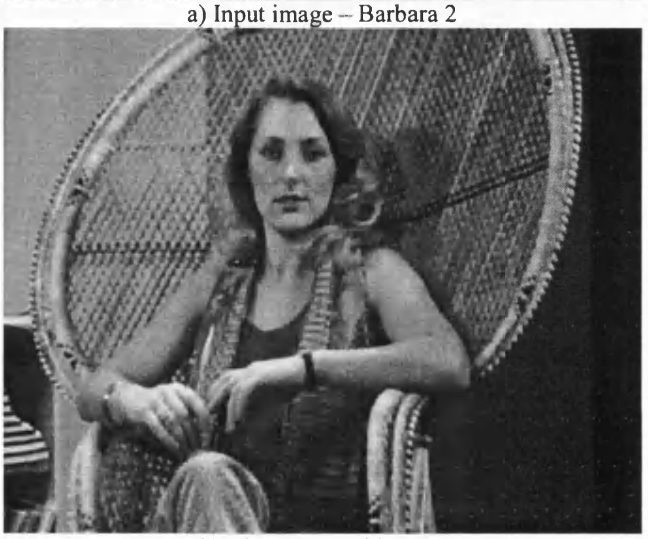


b) Closed image.

c) Opened image.

Figure 4.11: Basic morphological operations, closing and opening applied to the 720 x 576, 8bit greyscale Barbara 2 image.

a) Input image – Barbara 2



b) Close-Opened image.

c) Open-Closed image.

Figure 4.12: Basic morphological operations, close-open and open-close applied to the 720 x 576, 8bit greyscale Barbara 2 image.

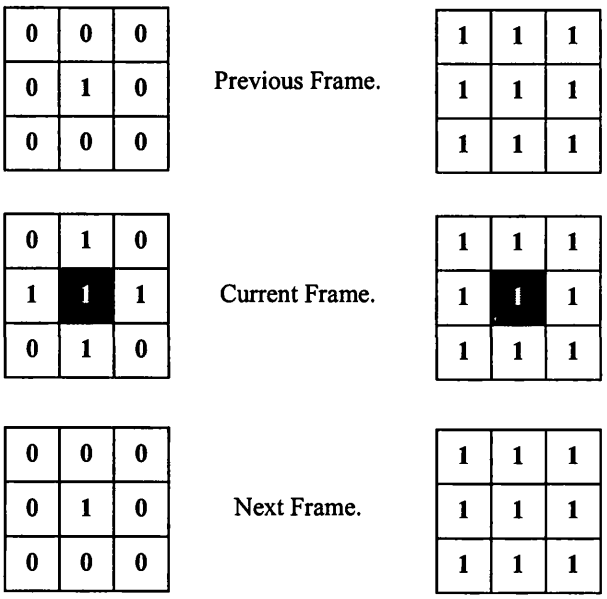


Figure 4.13: Extension of the 4nn (left) and 8nn (right) to 3D giving the 6nn (or 4_{nn}^{3D}) and 26nn (or 8_{nn}^{3D}) respectively.

4.4.2 3D Erosion

Again, erosion can be redefined in the same way to give 3D erosion as defined in equation 4.37. The other filters, open, close, etc. are still the same as before, but just use the above definitions for 3D (see equations 4.26 and 4.27).

$$A \ominus B = \min_{(i,j,k) \in B} \left(A_{x+i,y+j,z+k} \right) \tag{4.27}$$

4.4.3 Application to Video Preprocessing

These filters can now be applied to video sequences. Visually the results appear almost the same as for the 2D method. Figure 4.15 illustrates how similar the methods are by showing dilation on an image and on a sequence. However, 3D processing can remove more noise and smooth the images more because of the correlation between frames. This is good for compression as high frequency information is removed. However, this only works well when there is little motion in the sequence (see Figure 4.16). This can cause Dropouts (a dark spot) and Spikes (a bright spot) in the sequence. These methods posses the properties required for preprocessing, but need to be enhanced to overcome this limitation.

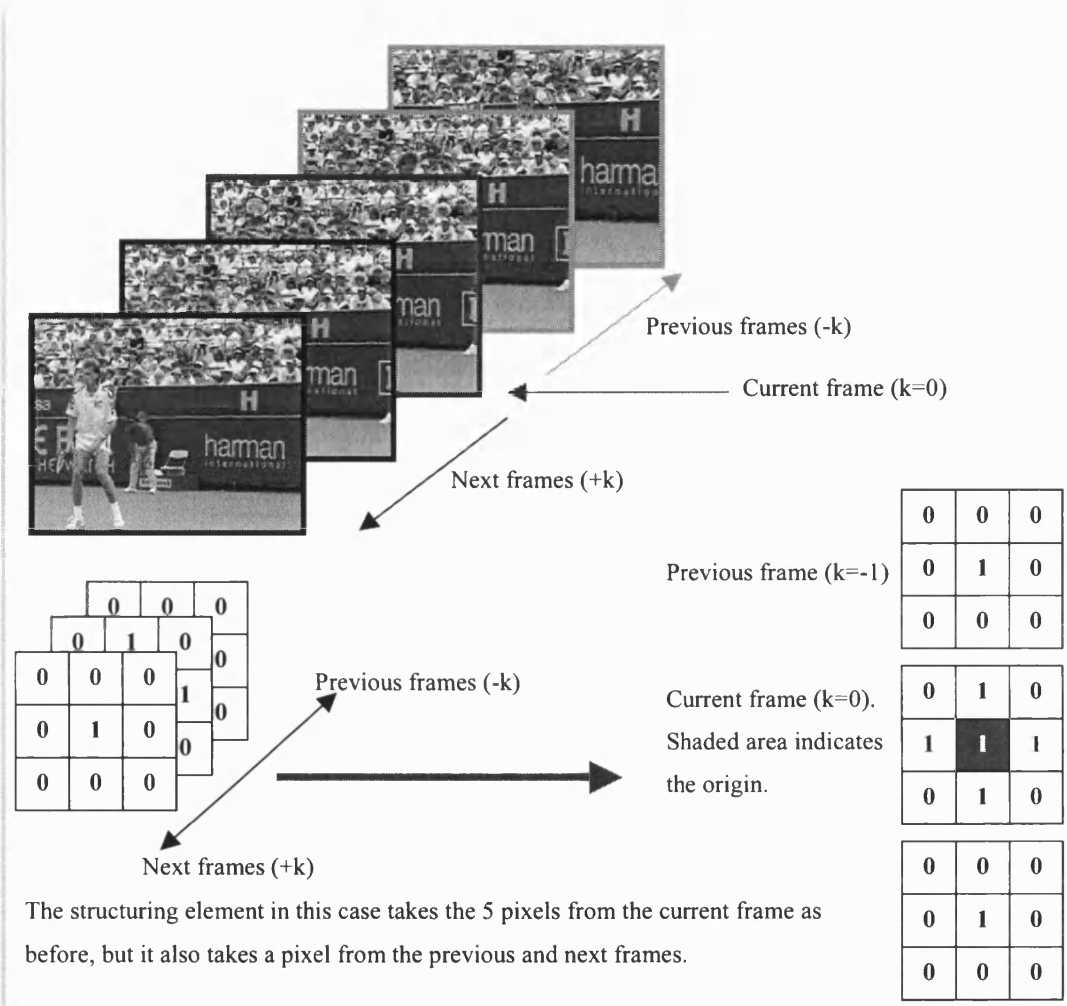
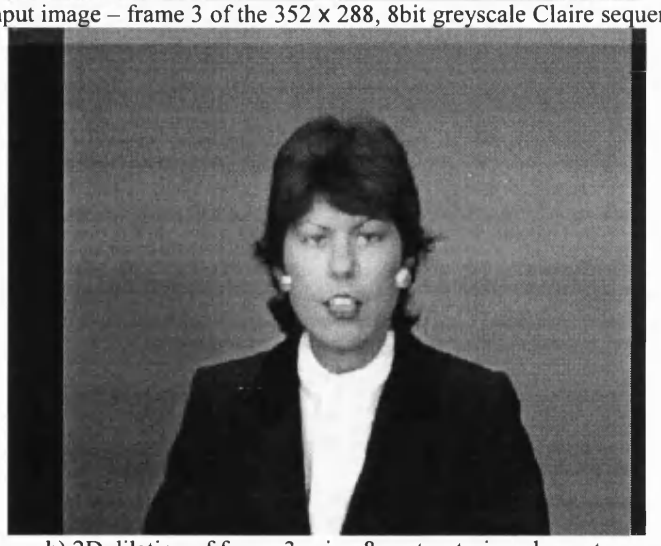
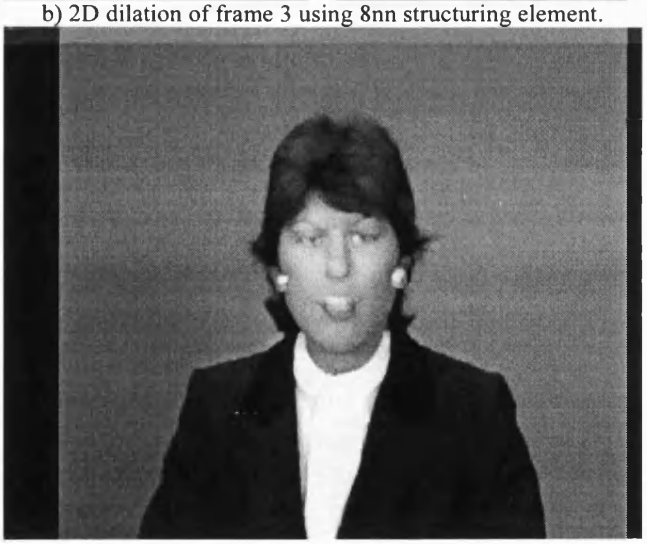


Figure 4.14: Basic principles of 3D morphology.

a) Input image – frame 3 of the 352 x 288, 8bit greyscale Claire sequence.

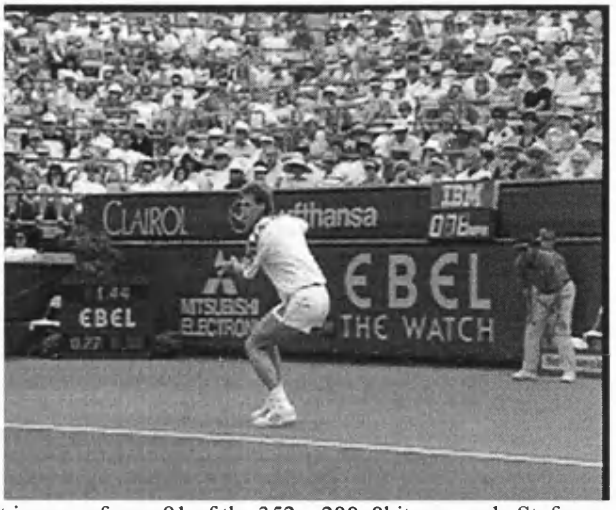


b) 2D dilation of frame 3 using 8nn structuring element.

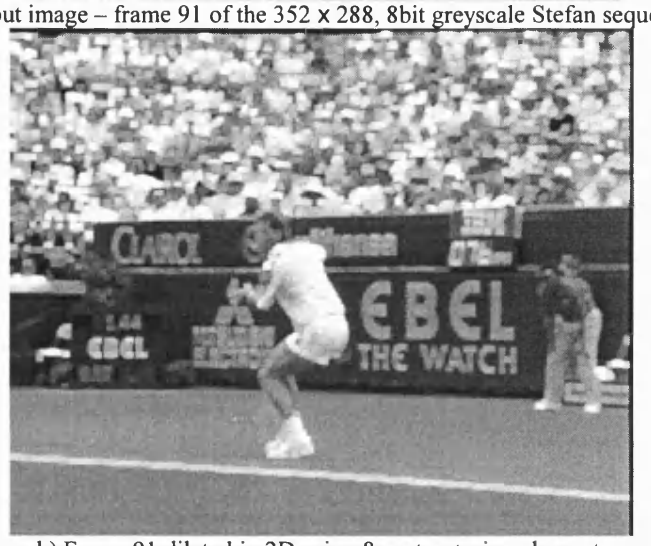


c) 3D dilation of frame 3 using 8^{3D}_{nn} structuring element.

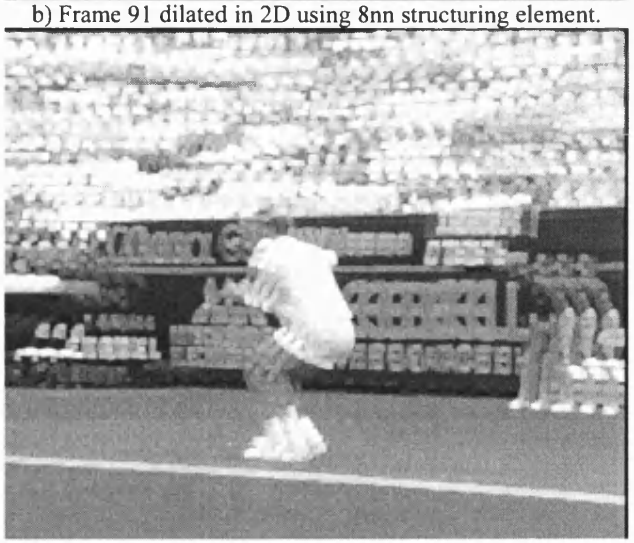
Figure 4.15: Example of the effects of 3D dilation on low motion video.



a) Input image – frame 91 of the 352 x 288, 8bit greyscale Stefan sequence.



b) Frame 91 dilated in 2D using 8nn structuring element.



c) Frame 91 dilated in 3D using 8^{3D}_{nn} structuring element.

Figure 4.16: Example of the effects of 3D dilation on fast motion video.

4.5 Granulometries

Granulometries were first introduced in the 1960's by Matheron for studying porus media. The basic idea is to progressively filter a set with an opening using structuring elements of increasing size (see Figure 4.17). The principle behind this is; *Let* $\Psi = (\psi_\lambda)$, $\lambda > 0$ be a family of image transforms dependant upon the parameter λ. For example ψ_λ could be an opening whereby the width of the structuring element is given by λ. This family constitutes a granulometry if and only if the following conditions are met:

- $\forall \lambda \geq 0$, ψ_λ is increaing,
- $\forall \lambda \geq 0$, ψ_λ is antiextensive,
- $\forall \lambda \geq 0,\ \mu \geq 0 \quad \psi_\lambda \psi_\mu = \psi_\mu \psi_\lambda = \psi_{\max(\lambda,\mu)}$.

The last point above, known as the 'semi group' property, implies that for every $\lambda \geq 0$, ψ_λ is an idempotent transform:

$$\psi_\lambda \psi_\lambda = \psi_\lambda \tag{4.28}$$

Matheron also proposed the following characteristics for a granulometry based on the above properties; A family $\Psi = (\psi_\lambda)$, $\lambda \geq 0$ is a granulometry if and only if it forms a decreasing family of openings such that:

- $\forall \lambda \geq 0$, ψ_λ is an opening,
- $\forall \lambda \geq 0,\ \mu \geq 0,\ \lambda \geq \mu \Rightarrow \psi_\lambda \leq \psi_\mu$

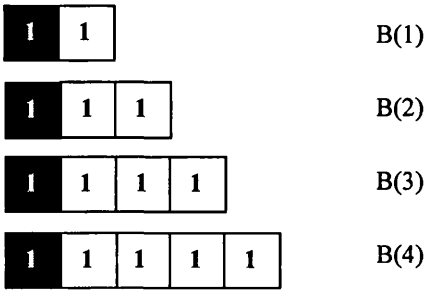


Figure 4.17: Increasing size structuring elements with the origin shaded.

Recall that an opening is idempotent, increasing and anti-extensive and hence can form a granulometry. It is not required that the openings are standard morphological openings, for example an algebraic opening may be used. This type of granulometry is often referred to as a granulometry by opening. Thus a binary and greyscale granulometry by opening can easily be defined as:

$$\alpha_r(X) = X \circ B(r) \tag{4.29}$$

where X is the set or image being operated on, B is the structuring element and r is the current scale or size of the structuring element. If all α_r are translation invariant , and scale compatible, then it is called '**Minkowski Granulometry**'. As with basic morphological operators, there is also a dual to this, the '**granulometry by closing**'. This is given by the definition; an increasing family $\Phi = (\phi_\lambda),\ \lambda \geq 0$ of closings that is such that $\forall \lambda \geq 0, \mu \geq 0,\ \lambda \geq \mu \Rightarrow \phi_\lambda \geq \phi_\mu$, is a granulometry by closing. Recall that this is true as a closing is also idempotent, increasing and extensive. Thus a binary and greyscale granulometry by closing can easily be defined as:

$$\beta_r(X) = X \bullet B(r) \tag{4.30}$$

It can be seen from the increasing structuring elements in Figure 4.17, that as the structuring elements increases, progressively larger objects, or granules' will drop through. For this reason, a granulometry is often compared to a sieve where by a fine sieve is used initially with progressively more coarser sieves being used. The typical sieve structure, regardless of the domain (i.e. binary or greyscale) and dimension (i.e. 1D, 2D, etc.) is shown in Figure 4.18. shows a simple 1D example of a sieve using the structuring elements from Figure 4.17. Note that the output will depend on the location of the origin within the structuring element and if an Open-Close is used or a Close-Open. Origins of the structuring elements are shaded. Since this method produces images at progressively increasing scales, this can be classified as a Scale Space and also as an ASF (see section 4.6.2) [64], [68] - [74]. Almost any filter can be used in the sieve structure, so for example instead of an opening or closing, a simple erosion or dilation may be used or an Open-Close. However, it should be noted that some operations, whilst they may work in the sieve structure, cannot be called granulometries. There are several applications of this structure, ranging from compression to feature extraction. One interesting application measures the rate at which the data is sieved, which provides a signature of the data with respect to the granulometry used. This is called '**Pattern Spectra**' of which there are many different interpretations and implementations of this [75].

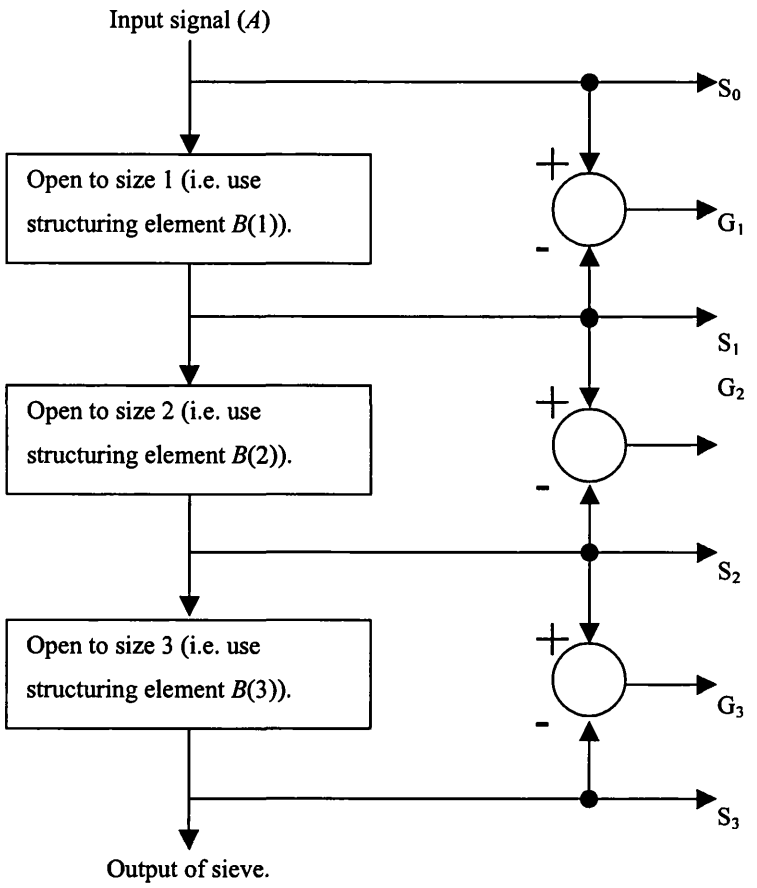


Figure 4.18: The sieve structure.

Both 2D and 3D data can be filtered using the same structure. However, unlike 1D, there is a much more varied choice of how to increase the structuring element. It is difficult to increase a structuring element by 1 pixel every step. For a structuring element of size 2, there are 8 possible structuring elements. Structuring elements are usually increased along one of their dimensions. For example, a box structuring element may be increase 3 x 3, 5 x 5, etc (see Figure 4.20). Alternitavly a disk structuring element may be used and its radius increased by 1 at every scale. Although these structuring elements still form a granulometry, the increasing structuring element can still give a coarse output. For example, increasing the boxstructuring element from its initial 3 x 3 to 5 x 5 increases the area of the objects to be removed from 9 to 25. Not only is this a large increase, there is also the problem that thestructuring element look for particular shapes within the image, hence prior knowledge about the image must be known. The only way of increasing the scales by one element at a time is to use AM (see section 4.6.1).

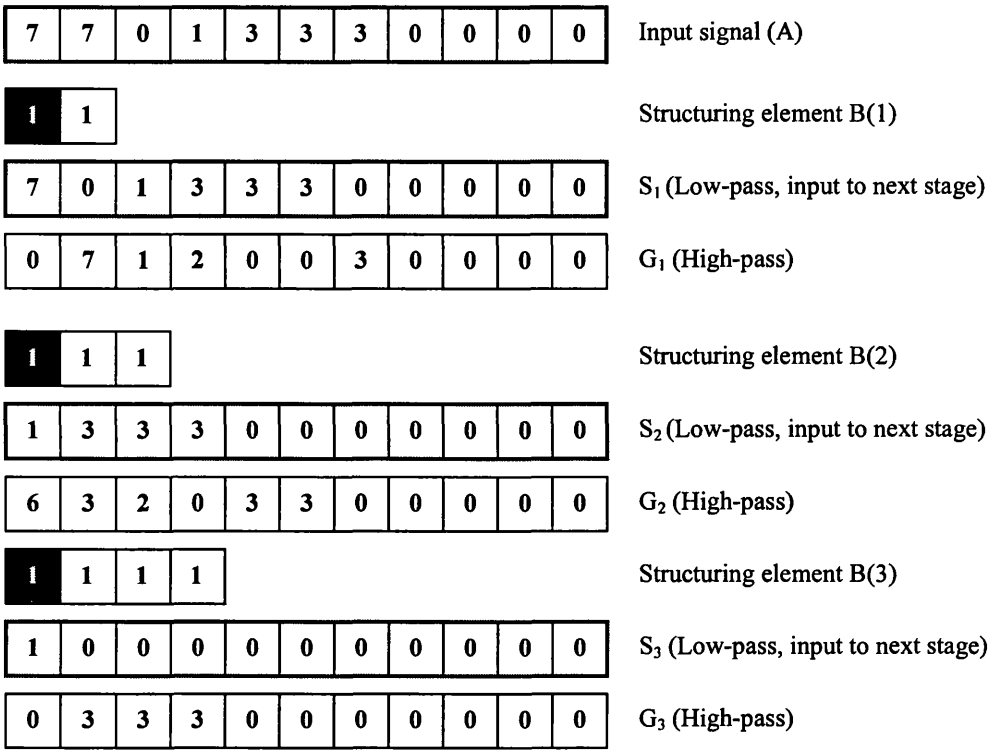
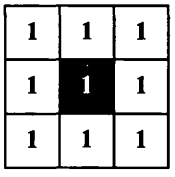
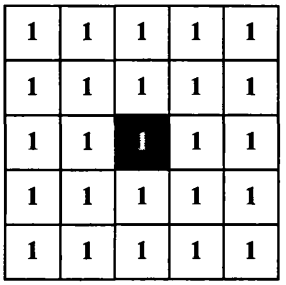


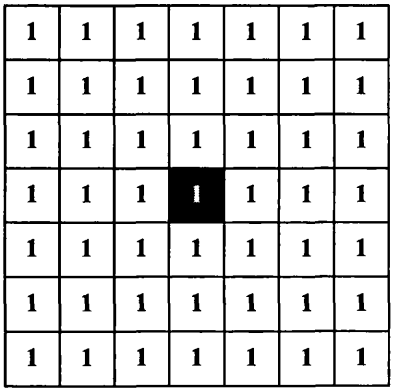
Figure 4.19: Sieve example of size 2.



a) 3 x 3 structuring element.



b) 5 x 5 structuring element.



c) 7 x 7 structuring element.

Figure 4.20: Increasing square structuring element.

4.6 Attribute Morphology

The morphological operators discussed so far operate on the entire image and have the same problem as that of the mean and median. That is, apart from the shape and size of the structuring element, there is no other way to control the filtering process. This is made more difficult using the sieve structure in higer dimensions as the size of the structuring element is increased in successively larger steps. Often only part of the image is required to be filtered. This both increases efficiency as only part of the image is filtered, and it also retains more of the image detail.

4.6.1 Area Morphology

Cheng and Venetsanopoulos [76] developed two adaptive morphological filters that operate only on parts of the image. Cheng and Venetsanopoulos called their operators, an Opening Operator (NOP) and a Closing Operator (NCP). Vincent used the same operators but called the NOP an Area-Opening (AO) and the NCP an Area-Closing (AC) [77], [78]. Consequently the study of this topic became known as AM. These operators operate in a similar way to the granulometry. A family of structuring elements, A_λ is defined such that λ is the area size to use. For example if λ is 2, then only the origin and one other connected pixel are included in the structuring element as shown in Figures 4.21 and 4.22 for 4nn and 8nn connectivity respectivley. Using this definition for A_λ, the area opening operation can thus be defined as:

$$\gamma_\lambda(X) = \bigvee_{S \in A_\lambda} (X \circ S) \tag{4.31}$$

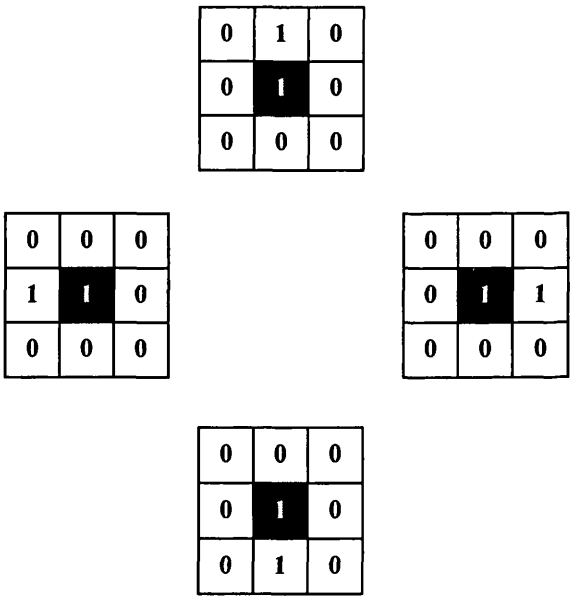


Figure 4.21: Example of A_λ for 4nn and λ =2. Shaded cell is the origin.

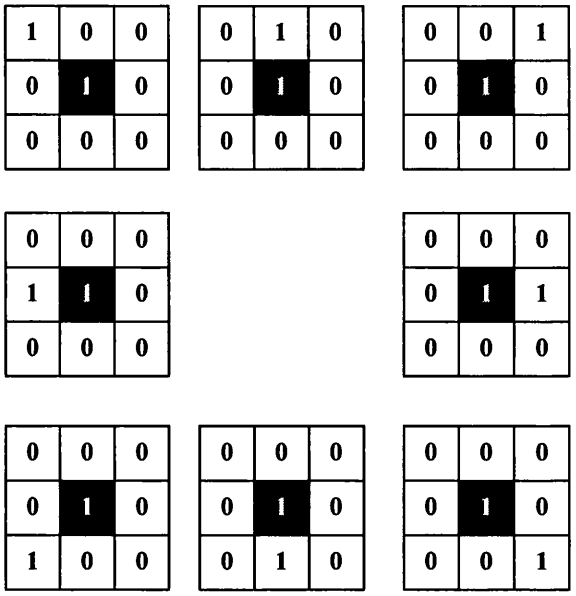


Figure 4.22: Example of A_λ for 8nn and λ=2. Shaded cell is the origin.

Essentially all this is doing is taking the maximum (or union) of all the possible openings using each possible structuring element of the family B^N. The max of a family of openings is still an algebraic opening. Conversely the area closing is given by:

$$\varphi_\lambda(X) = \bigwedge_{S \in A_\lambda} (X \bullet S) \tag{4.32}$$

This works in almost the same way as the area opening, except that the minimum (or intersection) of all of the possible closings is obtained. Again the min of a family of closings is still an algebraic closing. This method is actually picking the result that obtains the best result from each structuring element; in essence the ideal structuring element is used at each location. The more common and easier way of interpreting this is to completely ignore the structuring element together and rather think of an adaptively growing structuring element, which adapts to the characteristics of the image being filtered.

4.6.2 Filter Structures

Like the basic morphological operators, these operators may also be combined together. However there are two distinct ways that this may be done. The first method works the same as the basic opening and closing. Given two operators, α, ψ with the attribute λ, one operator is place after the other forming and AF as given by:

$$AF_\lambda(I) = \alpha_\lambda(\psi_\lambda(I)) \tag{4.33}$$

For AM, this could produce an Area-Open-Close (AOC) or an Area-Close-Open (ACO) as given in equations 4.34 and 4.35.

$$AOC_{\lambda}^{AF}(I) = \varphi_{\lambda}(\gamma_{\lambda}(I)) \tag{4.34}$$

$$ACO_{\lambda}^{AF}(I) = \gamma_{\lambda}(\varphi_{\lambda}(I)) \tag{4.35}$$

This just opens (repectivly closes) the image to the given area size and then closes (repectivly opens) to the given area size. The second method is derived from placing an area operator into the sieve structure (see section 4.5) or scale space [64]. As images are filtered through the sieve, granules fall out as before, but this technique has two distinct advantages over those described previously:

- Both light and dark points are filtered at each level,
- The granules increase by 1 pixel at a time, thus less detail is lost at each stage.

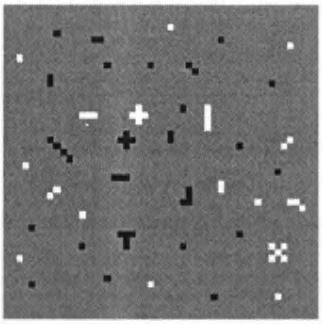
This is shown more clearly in Figure 4.23. Given two operators, α, ψ with the target attribute of λ, then the sieve structure, or more commonly referred to as an ASF is given in equation 4.36. Again the area operators can easily be placed into this to give for example, an ASF version of the AOC which is given by equation 4.37. Any function can be placed into these AF and ASF structures. There are some differences in the performance of the two, which is investigated further in chapter 7.

$$ASF_{\lambda}(I) = AF_{\lambda}(...(AF_{\lambda-1}...(AF_1(I)))) = \alpha_{\lambda}(\psi_{\lambda}(...\alpha_{\lambda-1}(\psi_{\lambda-1}(...\alpha_1(\psi_1(I))..))..)) \tag{4.36}$$

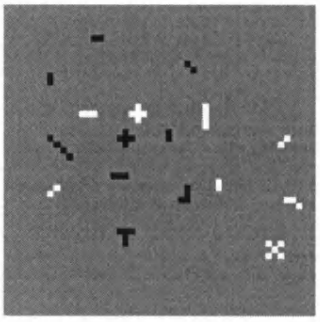
$$AOC_{\lambda}^{ASF}(I) = \varphi_{\lambda}(\gamma_{\lambda}(...\varphi_{\lambda-1}(\gamma_{\lambda-1}(...\varphi_1(\gamma_1(I))..))..)) \tag{4.37}$$

4.6.3 Generalisation to Other Attributes

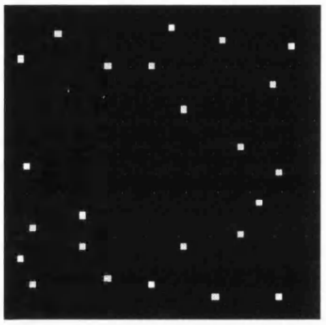
Breen and Jones generalised AM so that any attribute, for example such as volume, complexity, motion or power, could be used instead of area [79] - [83]. This method is generally reffered to as '**Attribute Morphology**', which AM is a subset of. This works on the same principle as that described for area morphology, except that instead of growing the structuring element until it reaches a predetermine size, an attribute of the pixels in the structuring element is measured. For example, for contrast the difference between the highest and lowest valued pixel within the structuring element is measured. If this attribute is greater than some threshold, then that pixel is left untouched. There are again several applications for attribute morphology. Sun and Wu use the size and shape attribute to segment fiducial marks [52].



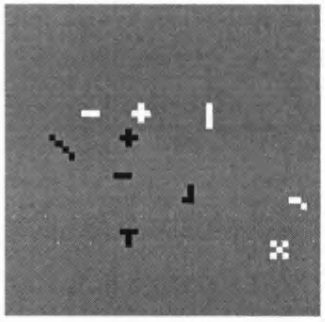
Input image (S_0)– Artificial test image



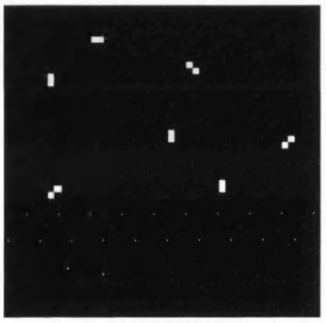
Low-pass filtered (S_1)



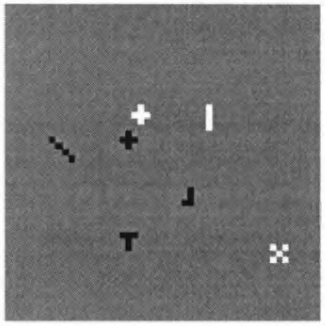
High-pass filtered – Granules of size 1(G_1)



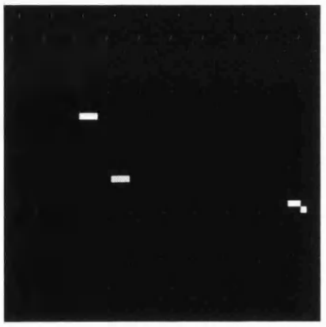
Low-pass filtered (S_2)



High-pass filtered – Granules of size 2(G_2)



Low-pass filtered (S_3)



High-pass filtered – Granules of size 3(G_3)

Figure 4.23: 2D sieving example using an artifitial image.

4.6.4 Application to Video

Attribute morphology can be applied to images and sequences as shown above and will cope better with motion in sequences, as the adaptive structuring element will actually track objects. Chapter 9 develops this idea further and addresses some of the problems of this method. It should also be clear that provided no data is thrown away, then output of the sieve can be combined to reconstruct the original image. This could be used to form a scale space CODEC.

4.7 Conclusion

This chapter has introduced the basic principles of mathematical morphology. AM and attribute morphology have been introduced, which are developed further in chapters 7, 8 and 9 for application to psychovisually lossless preprocessing digital images and noise reduction for digital video. Although this thesis deals mainly with greyscale images, mathematical morphology can also be used for colour image processing **[84] - [86]**. However, the method for the application of mathematical morphology to colour images is still an issue. For example the mathematical morphology operators can be applied to each colour componend seperatly or linked together using vectors and vector operators **[84] - [87]**.

5 Noise in Digital Images

Most images and video sequences, unless they are Computer Generated Imagery (CGI), will contain noise in some form. Noise can be introduced at various stages in an imaging system, such as the capture, processing or transmission of the signal. The visibility of noise within an image depends on many factors such as the type of noise, the intensity of the noise and the image content. Thus although many images may look noise free, they may still contain some noise that is not visible by the HVS (see chapter 3). In some cases this information can have a significant impact on the performance of an image compression system. This chapter looks at the different types of noise commonly found (see section 5.1), how to estimate the amount of noise present in an image with no prior information (see section 5.2). Noise reduction methods are then described in section 5.3, including a review of recently developed ideas. Finally section 5.4 describes how each of these elements may be used to develop a preprocessing system.

5.1 Sources of Noise in Digital Video

Noise can easily be modelled by using a Probability Density Function (PDF) [88]. This describes the probability that a pixel is in error based upon the type of noise being modelled. In most cases noise is modelled as white noise, meaning that the noise has a constant power spectrum (i.e. it has constant intensity irrespective of the frequency). White noise is used as it is considered to be a worst-case approximation. When a signal is transmitted through some medium, through a Printed Circuit Board (PCB), cabling or broadcasting, noise that is independent of the signal is usually added to it. The cause of this could be imperfections in the PCB, interference from other broadcasts, from the image sensor (i.e. Charge Couple Devices or CCDs), which produce narrow band (moiré) thermal white Gaussian noise, etc. This form of noise is often called Additive Noise as given by:

$$r = s + n \tag{5.1}$$

where s is the signal and n is the noise. However in some cases, the noise intensity depends upon the signal intensity. This is called Multiplicative Noise and is given as:

$$r = s + sn \tag{5.2}$$

where s is the signal and n is the noise. Equation 5.2 is often simplified and given as:

$$r = sn \tag{5.3}$$

Examples of this form of noise include, the grain of photographic film, speckle noise, which is generated from imaging systems using coherent radiation such as used in Synthetic Aperture Radar (SAR) and ultrasound imaging. There are several noise models used as described in the following

sections. Even the basic operation of sampling an analogue signal by using an ADC can cause quantisation noise [9], [89]. The noise can either be in the form of not enough bits being used to represent the image or by the signal being converted incorrectly leading to some of the data being placed at the incorrect levels. Four of the most common noise models found in image processing are described here. There are however more types of noise such as Gamma and Rayleigh [90]. All of the noise models discussed are found in analogue and digital systems at various stages. For example, Gaussian noise could be used to model noise present on a CCD or impulse noise may be introduced when the data is being transmitted. Hence all four types of noise may be present in both the analogue and digital systems.

5.1.1 Gaussian Noise Model

Gaussian noise, also called the normal, is a good approximation to many practical instances. Gaussian noise is usually applied uniformly over the entire image, is zero-mean and is applied using the additive model (see equation 5.1). This is commonly referred to as Additive White Gaussian Noise (AWGN). This type of noise is often considered to be the worst case possible. The Gaussian PDF is given as:

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) \tag{5.4}$$

where μ is the mean value, σ is the standard deviation and σ^2 is the variance. Using this PDF, 70% of the values will lay within the range of $[(\mu-\sigma),(\mu+\sigma)]$ and 95% will lay within the range $[(\mu-2\sigma),(\mu+2\sigma)]$. The PDF and a degraded image using Gaussian noise is shown in Figure 5.1. Gaussian noise is often used to approximate noise that may be present on the output of the CCD and so this noise would be present in both analogue and digital systems.

5.1.2 Impulse Noise

This type of noise corrupts individual pixels by significantly changing their intensity with respect to their neighbours. This type of noise is usually found within digital storage and transmission systems. This type of noise has several options. To start with noise can be either added to a pixel increasing its value or subtracted decreasing its value. This is known as Bipolar Impulse Noise. This type of noise works by saying for example that there is a 15% chance that the noise will be positive (P_P) and say 5% will be negative noise (P_N). Each pixel (p) in the image is then processed using:

$$p=\begin{cases} p+n & \textit{with a probability of } P_P \\ p-n & \textit{with a probability of } P_N \\ p & \textit{otherwise} \end{cases} \tag{5.5}$$

If either P or P_N are 0, then noise will only be added or subtracted but not both. This is known as Unipolar Impulse Noise. A degraded image using Impulse noise is shown in Figure 5.1. Salt and Pepper Noise is a special case of impulse noise, which just saturates the pixels. For example in an 8bit image, any pixel that is to be corrupted, instead of adding a noise value to it, the pixel is completely replaced with either 255 (salt) or 0 (pepper). Hence equation 5.5 becomes:

$$p = \begin{cases} n_{salt} & \textit{with a probability of } P_P \\ n_{pepper} & \textit{with a probability of } P_N \\ p & \textit{otherwise} \end{cases} \tag{5.6}$$

where n_{salt} is the maximum saturated value (i.e. 255) and n_{pepper} is the minimum saturated value (i.e. 0). This type of noise may be present in both analogue and digital systems. For example impulse noise may the result of interference whilst being transmitted. However the effect is completely different between the two systems. An analogue system will result in the image as shown in Figure 5.1 where the image values are directly affected. A digital system will result in the decoded image containing artefacts or even unusable data. This is because the actual bit stream would be altered.

5.1.3 Exponential Noise

Exponential noise is a special case of Erlang noise with b set to 1. The PDF for this form of noise is:

$$p(z) = \begin{cases} a\ \exp(-az) & \textit{for } z \geq 0 \\ 0 & \textit{for } z < 0 \end{cases} \tag{5.7}$$

where $a>0$. The mean and the variance are given by equations 5.8 and 5.9 respectively. The PDF and a degraded image using Exponential noise is shown in Figure 5.2.

$$\mu = \frac{1}{a} \tag{5.8}$$

$$\sigma^2 = \frac{1}{a^2} \tag{5.9}$$

5.1.4 Uniform Noise

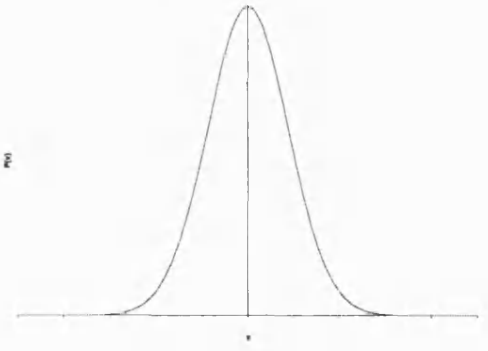
Uniform noise adds noise to an image so that the noise level is uniform across the desired range. The PDF for this type of noise is given in equation 5.10 and the mean and variance are give by equations 5.11 and 5.12 respectively. The PDF and a degraded image using Uniform noise is shown in Figure 5.2.

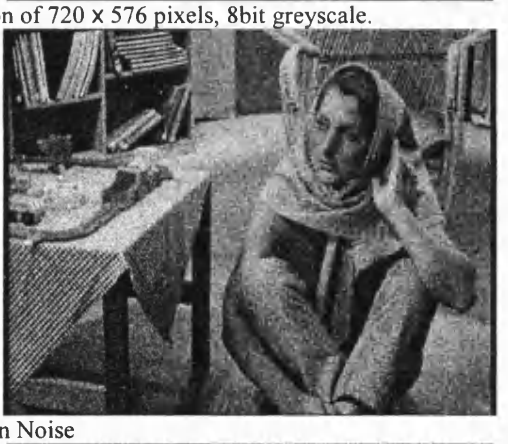
$$p(z) = \begin{cases} \frac{1}{b-a} & for\ a \le z \le b \\ 0 & \text{Otherwise} \end{cases} \tag{5.10}$$

$$\mu = \frac{a+b}{2} \tag{5.11}$$

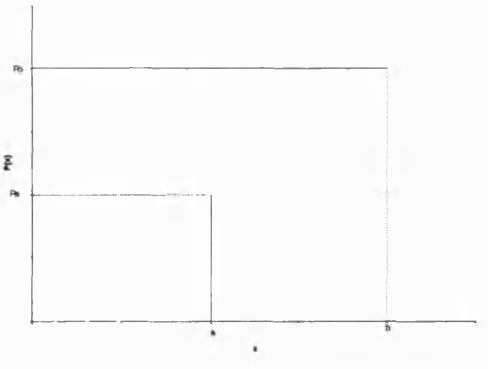
$$\sigma^2 = \frac{(b-a)^2}{12} \tag{5.12}$$

a) Input Image, Barbara, with a resolution of 720 x 576 pixels, 8bit greyscale.





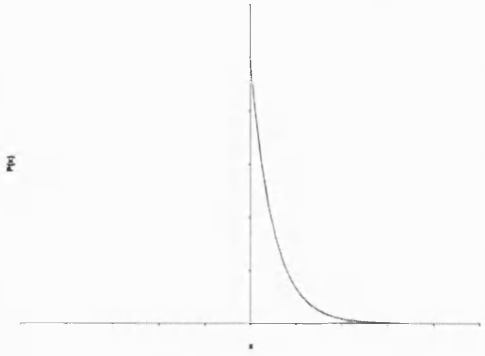
b) Gaussian Noise

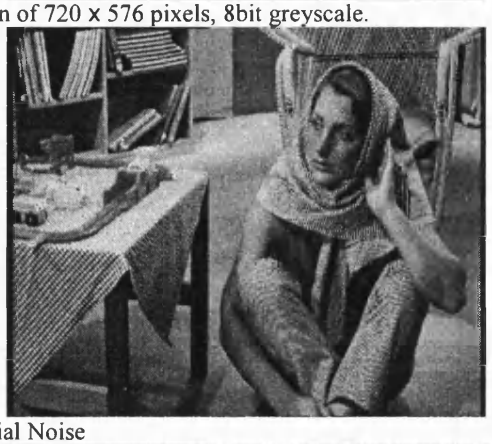


c) Impulse Noise

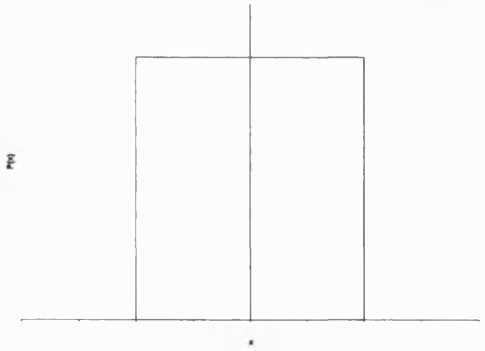
Figure 5.1: Gaussian and Impulse noise model PDF (left) and resultant image (right).

a) Input Image, Barbara, with a resolution of 720 x 576 pixels, 8bit greyscale.





b) Exponential Noise



c) Uniform Noise

Figure 5.2: Exponential and Uniform noise model PDF (left) and resultant image (right).

5.2 Noise Estimation

In the real world, corrupted images would be presented for filtering with little or no prior knowledge. For example, the type of noise may be know, but the amount of noise present usually will not be known. Many noise reduction systems employ an estimator that can give a good approximation to the amount of noise present. These vary from simple block based estimators to more complex parallel algorithms and video specific estimators [91] - [93]. One of the most simplistic methods is to filter the noisy image, for example using the mean or median, and measure the difference between the filtered image and the original. This is usually constrained by either using pixels that do not belong to edges or by using blocks that have a variance below a predetermined threshold. However, this would

need a large training set to determine the best filter and attributes, such as mask size and weights, to use and the value to use for any thresholds. In addition, there is the problem of prefiltering may remove some of the data that should be used when calculating the noise level, and also adds an extra level of complexity to the overall system.

The method that is used for the work in this thesis is a simple block based method, whereby the noisy image is split into non-overlapping blocks [91]. The variance, as given in equation 5.13, is calculated for each block. The smallest n percent of these blocks are then averaged together to give an estimate of the variance of the noise present in the image. However, this method still requires a block size and percentage to be specified. Sections 7.1.3 and 9.5 use this technique to apply the preprocessing system to unseen data and investigates the best attribute values to use.

$$\text{variance}(X) = \frac{\sum_{i=0}^{N-1}(x_i - \bar{x})^2}{N-1} \tag{5.13}$$

where N is the number of elements in the signal and $\bar{x}$ is the mean of the signal given by:

$$mean(X) = \frac{\sum_{i=0}^{N-1} x_i}{N} \tag{5.14}$$

5.3 Noise Reduction

One of the goals of image preprocessing is to form an approximation of the original image having been given a corrupted version of that image. There are several methods that can be used for noise reduction such as linear, non-linear, spatial and frequency methods. Noise usually appears in images as discrete isolated pixel variations that are spatially and temporally uncorrelated. Adjacent pixels in an image and adjacent frames in a sequence are often correlated which makes it easier to detect noise.

5.3.1 Linear Filtering

Since noise can be classified as high frequency information, simple low-pass filters such as the mean can be used to efficiently remove high frequency information. The mean filter works by using the convolution equation given by:

$$G(j,k) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} F(m,n)H(m-j-C, n-k+C) \tag{5.15}$$

where M is the width of the image, N is the height of the image, H is the impulse response array (also called the mask), F is the input image and C is given by equation 5.16.

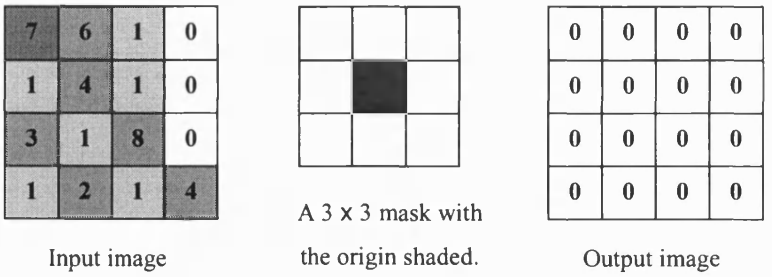
$$C = \frac{L+1}{2} \tag{5.16}$$

where L is the size of the impulse response array. For example, using a 3 x 3 mask, $L=3$. The variable C stops the mask from going outside of the image boundaries. This has the disadvantage that each application reduces the size of the image. The more commonly used method is to set $C=0$, and to reflect the image. The mask for a 3 x 3 mean filter is given by equation 5.17. This is a special case of a 3 x 3 parametric low-pass filter, which has the mask defined in equation 5.18. The mask decides how much of the local information to use when calculating the output for a given pixel. The mask may be of any size; however the computational efficiency is degraded as the mask size increases. Also the image becomes more blurred as the size increases. More commonly, the mean can be thought of as a window that slides across the image calculating the average value of the pixels that lay within the mask boundaries as shown in Figure 5.3.

$$H = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5.17}$$

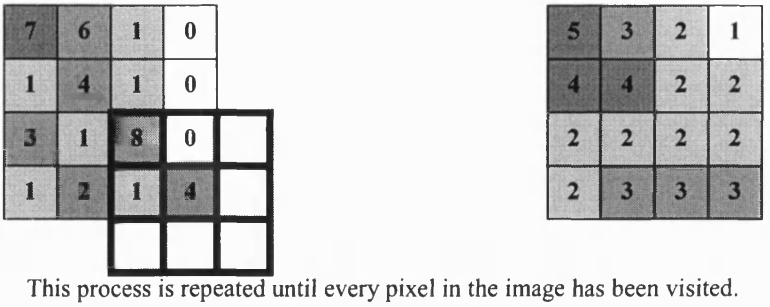
$$H = \left(\frac{1}{b+2}\right)^2 \begin{bmatrix} 1 & b & 1 \\ b & b^2 & b \\ 1 & b & 1 \end{bmatrix} \tag{5.18}$$

Figure 5.4 shows an example of impulse noise removal using the mean filter. The original image has been corrupted with 10% impulse noise, i.e. 10% of the image became noise, which when compared with the original image, gives a PSNR (see section 3.2.1) of 15.89dB. Using a window size of 5 x 5 produces the best result for this particular image, resulting in a PSNR of 21.45dB, an improvement of 5.56dB. Although the PSNR shows a significant gain in quality, it can be seen that there has been some degradation in the quality of the image, which becomes more apparent with larger window sizes.



The mask is placed over the first pixel. The average value of the pixels within the mask is then calculated and rounded if needed, in this case 4.5 becomes 5, which is then put in the corresponding pixel in the

The mask is then slide along to the next pixel and the average value calculated for the pixels laying within the mask.



This process is repeated until every pixel in the image has been visited.

Figure 5.3: Image filtering using a mask.

a) Input image - Barbara

b) Barbara corrupted with 10% impulse noise.

c) Barbara filtered with the mean using a 5 x 5 window.

d) Barbara filtered with the mean using a 15 x 15 window.

Figure 5.4: Impulse noise filtering using the mean filter and the 720 x 576, 8bit greyscale Barbara image.

Figure 5.5 shows results from filtering Gaussian noise using the mean filter. A noisy Gaussian image was produced using a $\sigma = 24.7$, which when compared with the original image, has a PSNR of 20.45dB. The best mean filter, using a window size of 3x3, produces a PSNR of 24.77dB. It can be seen that this filter significantly alters high frequency components and also introduces values that are not present in the original image. This results in the image being blurred and thus object boundaries can become obscured. Generally, the mean filter is used for Gaussian noise rather than impulse type noise.

5.3.1.1 Homomorphic Filtering

Homomorphic filtering [94] is useful for removing multiplicative noise. This works by assuming the multiplicative model from equation 5.3 and takes the logarithm of the input image to obtain an additive version as given by equation 5.19. Simple linear filtering techniques such as the mean can then be applied to remove the noise. The exponential of the output is then taken to get the image back into the original domain. This is shown more clearly in Figure 5.6.

$$\log(r) = \log(s) + \log(n) \tag{5.19}$$

a) Input image - Barbara

b) Barbara corrupted with 20dB Gaussian noise.

c) Barbara filtered with the mean using a 5 x 5 window.

d) Barbara filtered with the mean using a 15 x 15 window.

Figure 5.5: Gaussian noise filtering using the mean filter and the 720 x 576, 8bit greyscale Barbara image.

5.3.1.2 Gaussian Smoothing

The Gaussian function can also be used to filter images. This can be thought of as a simple weighted filter where the weights are the Gaussian mask, which is given in equation 5.20. The mask, which is commonly shown as a 3D plot rather than a 2D mask (see Figure 5.7).

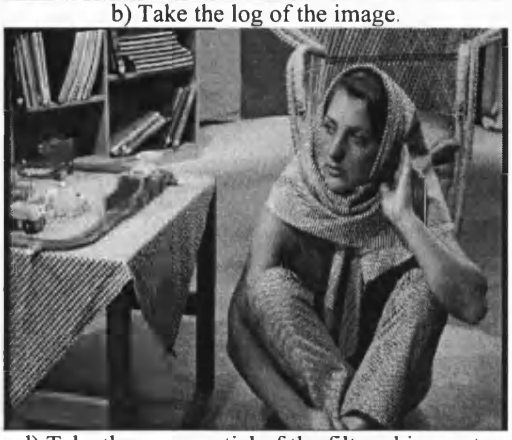
$$Gaussian(x,y) = \frac{1}{2\pi\sigma^2}\exp-\left(\frac{x^2+y^2}{2\sigma^2}\right) \tag{5.20}$$

where σ is the variance, which controls the width of the Gaussian function. Figure 5.8 shows an example of filtering impulse noise using the Gaussian filter. The best output was produced using a window size of 9 x 9 and a $\sigma = 1.1$. This resulted in a PSNR of 21.94dB, which is better than the mean, but still is not as effective as a median. Figure 5.11 shows an example of Gaussian noise removal using the Gaussian filter. The best output gave a PSNR of 25.72dB using a window size of 7 x 7 and a $\sigma = 0.7$. This is slightly better than the best mean output.

a) Input Image – Barbara.

b) Take the log of the image.

c) Use a linear filter to reduce the noise.



d) Take the exponential of the filtered image to get the final result.

Figure 5.6: Homomorphic filtering example using the 720 x 576, 8bit greyscale Barbara image.

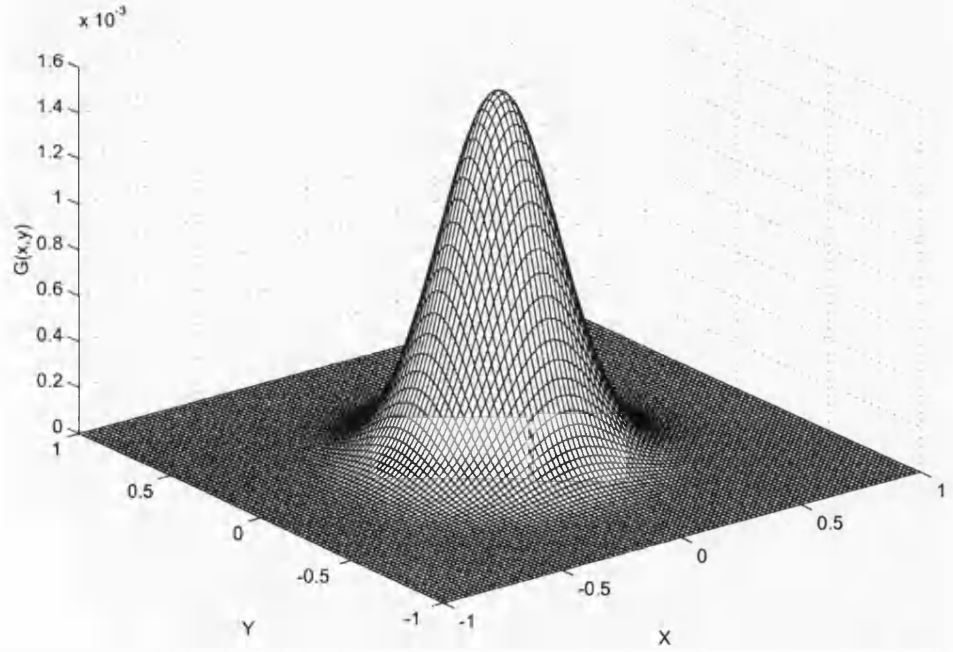


Figure 5.7: A Gaussian filter mask.

a) Input image - Barbara

b) Barbara corrupted with 10% impulse noise.

c) Barbara filtered with the Gaussian using a 5 x 5 window and $\sigma = 1$.

d) Barbara filtered with the Gaussian using a 15 x 15 window and $\sigma = 1$.

Figure 5.8: Impulse noise filtering using the Gaussian filter and the 720 x 576, 8bit greyscale Barbara image.

a) Input image - Barbara

b) Barbara corrupted with 20dB Gaussian noise.

c) Barbara filtered with the Gaussian using a 5 x 5 window and $\sigma = 1$.

d) Barbara filtered with the Gaussian using a 15 x 15 window and $\sigma = 1$.

Figure 5.9: Gaussian noise filtering using the Gaussian filter and the 720 x 576, 8bit greyscale Barbara image.

5.3.2 Non-linear Filtering

Linear filters perform well on images corrupted with continuous noise such as Gaussian and uniform noise. However when used with impulse type noise, whilst they remove the noise they often provide too much smoothing of other image features, especially edge details. Non-linear filtering provides a finer control over the amount of filtering and give much better preservation of edge information.

5.3.2.1 Median Filtering

One of the most widely used non-linear filtering methods is Order Statistics. A median filter is a member of this family and has been widely researched [47], [48], [95], [96]. The median of a sequence of sorted values $A = \{a_1, a_2, a_3, \ldots, a_n\}$ is given by equation 5.21.

$$median(A) = \begin{cases} A\left(\frac{n+1}{2}\right) & \text{if n is odd} \\ \frac{A\left(\frac{n}{2}\right) + A\left(\frac{n}{2}+1\right)}{2} & \text{if n is even} \end{cases} \tag{5.21}$$

Using a median filter partially overcomes the problems of the mean filter as it retains more of the original image detail and cannot introduce new image values so long as the mask size contains an odd number of pixels. In addition, because of these features the median preserves more of the edge detail than the mean. Figure 5.10 shows an example of impulse noise removal using the median filter. The image was corrupted with 10% impulse noise, i.e. 10% of the image became noise, which when compared with the original image, gives a PSNR of 15.89dB. The median filter gives a resultant PSNR of 23.53dB where as the mean only generated a PSNR of 21.45dB. As can be seen from Figure 5.10, the results produced by the median filter produce a much more pleasing image, in that more of the edge information is retained. Even with a large window size, 15 x 15 for example, the major features of the image can still be made out quite clearly whilst the mean filter has significantly blurred the image.

The more common Gaussian noise is shown using the mean and median filters in Figure 5.11. The noisy Gaussian image was produced using a $\sigma = 24.7$, which when compared with the original image, has a PSNR of 20.45dB. The best median filter uses a window size of 3 x 3 and has a PSNR of 24.04dB, which is worse than that produced by the mean filter, 24.77dB, thus showing that the mean filter is better at removing Gaussian noise whilst the median is better at removing impulse noise. In addition the visual difference between the mean and median filters are less noticeable. This has been confirmed by research done elsewhere [97], [98].

It can be seen that in both cases the median filter, despite removing a significant amount of the fine detail, still retains more of the edge information. However it can also be seen that the filter type and its parameters will depend upon the type of noise present and the amount of noise. Thus whilst the median filter is an ideal choice for impulse noise removal, it may not be the best choice for removing other types of noise. The median filter has been optimised in several different ways to achieve a low computational complexity [99], [100]. There are various improvements that can be applied to the median filter [101] such as using an adaptive mask size, the truncated median, weighted median, adaptive median [102], mean relaxed median [98], [103] - [105].

a) Input image - Barbara

b) Barbara corrupted with 10% impulse noise.

c) Barbara filtered with the median using a 5 x 5 window.

d) Barbara filtered with the median using a 15 x 15 window.

Figure 5.10: Impulse noise filtering using the median filter and the 720 x 576, 8bit greyscale Barbara image.

5.3.2.2 Outlier Filter

The outlier filter works by comparing each pixel to the average of its eight neighbours. If the difference is above a predetermined threshold the pixel is classified as a noisy pixel and is replaced by the average of its neighbours. The average of the eight neighbours for a 3 x 3 mask can be computed by using the convolution equation and the mask (see equation 5.15). The mask size can be any size. The main drawback to this method is determining a suitable threshold. If the threshold is too low, then image features will be removed as well as noise, if set too high then not all of the noise will be removed. Variations on this method have been developed by Davis and Rosenfeld **[106]**.

$$H = \frac{1}{8}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5.22}$$

a) Input image - Barbara

b) Barbara corrupted with 20dB Gaussian noise.

c) Barbara filtered with the median using a 5 x 5 window.

d) Barbara filtered with the median using a 15 x 15 window.

Figure 5.11: Gaussian noise filtering using the median filter and the 720 x 576, 8bit greyscale Barbara image.

5.3.2.3 Image Restoration

Image restoration is a method, which attempts to recover the original image from a degraded copy using prior knowledge about the degradation process. For example, the image may have been corrupted with noise, blurred, faded or a combination of these. The idea behind image restoration is to form an estimation ($\hat{X}$) of the input data (X) using all of the knowledge of how the degradation process works. Bayesian probability is one possible solution for this method. An estimate is found that maximises the probability:

$$\hat{X} = P(X \mid \text{data}, \text{model}) \tag{5.23}$$

This determines the probability of X given the degraded image data and a relevant degradation model.

5.3.3 Review of Current Image Noise Reduction Techniques

The mean and median filters are both simplistic methods of noise removal. More advanced and better filter exists such as the wiener filter or simulated annealing. Vasconcelos and Dufaux **[107]** present a preprocessing system specifically designed for low bit rate video CODEC's. Preprocessing is done in two stages using what the authors have called an adaptive low pass filter. The first stage has been called static prefiltering. Basically this section is designed to remove high frequency sensor (camera) noise. It has been designed specifically to avoid blurring boundaries whilst removing this noise. This is done in several steps:

1. Use the Sobel edge detector to find all edges in the image (this is called the edge map).
2. Use a morphological Open-Close (see section 4.1.3), to remove any noise from the edge map.
3. Select the most appropriate filter for the current pixel using this information (i.e. if a pixel is near or on an edge, then do not filter it). This uses the morphological open-close by reconstruction filter (see section 6.1).

The second stage of this method uses pairs of frames. At this point the authors assume that the HVS is not very sensitive to fast motion. They use the motion compensation from the CODEC to filter each part of the image according to the amount of motion as shown in the following steps:

1. Filter the input frame pair with an array of N low-pass filters with differing characteristics.
2. Select the best filter according to the amount of motion and prediction error.

This method has several problems. The properties of the HVS are assumed and have not been tested. The system was developed for use with Quarter Common Image Format (QCIF), which is 176 x 144 pixels, sequences at 10 fps using the H263 CODEC. No direct measurement in compression ratio or MSE is given so it is difficult to judge this method effectiveness.

Another approach is to try to identify and remove noise contained within each image. There are various ways to detect and remove noise. For example, Bayesian probability [108] can be used to decide if a pixel is noise. However, this can often be complicated, as various information about the image is required in order to generate a model to use. Two alternatives are now described [109], [110]. The first method is aimed at reducing noise from digitised material (i.e. analogue video that has been converted to digital) in real time using an Application Specific Integrated Circuit (ASIC) [109]. This assumes that there is a lot of impulse noise caused by the conversion and aims to remove just this noise. This is done by splitting the image into blocks for each frame (the authors have called these slices). This is similar to how MPEG uses the DCT. A group of these blocks, in the same position on different frames has been called a tube. To start with, two functions are used to calculate if there is noise or a scene change in the tube for the current slice by using:

- The absolute difference in luminance of two consecutive slices.
- The absolute difference in blue component of the two consecutive slices.

The temporal boundaries (how many past/future frames) are determined from these measurements. This is repeated for all the slices within the tube and the results are median filtered to form the tube. The entire sequence is processed in this way. Impulse noise is removed by using this method, but will be of little use if the source material is in a digital format (i.e. from a digital camera).

The second method is designed for use with MPEG-I **[110]**. This method uses the fact that MPEG groups images into different types (I, P or B frames). Each frame is filtered in a different way. For the I-frames, each pixel is filtered using the current and next frame using the simple equation:

$$\text{Filtered Image}_{(i,j)} = \beta\left(\text{Current frame}_{(i,j)}\right) + (1-\beta)\left(\text{Next frame}_{(i,j)}\right) \tag{5.24}$$

where β is the filter coefficient. For the P-frames, motion compensation from the CODEC is available. This uses the previous, current and next frames to give the filtered image as:

$$\begin{aligned}\text{Filtered Image}_{(i,j)} = {} & \beta(1-\alpha)\left(\text{Previous frame}_{(i,j)}\right) + \\ & \alpha\beta\left(\text{Current frame}_{(i,j)}\right) + (1-\beta)\left(\text{Next frame}_{(i,j)}\right)\end{aligned} \tag{5.25}$$

where α is another filter coefficient. This method will use either the previous motion compensated frame or the last filter frame instead of using the adjacent previous frame depending upon the error between these two frames and the current frame. The B-frames are filtered in a similar way using:

$$\begin{aligned}\text{Filtered Image}_{(i,j)} = {} & \beta(1-\alpha)\left(\text{Previous frame}_{(i,j)}\right) + \\ & \alpha\beta\left(\text{Current frame}_{(i,j)}\right) + (1-\beta)\left(\text{Next frame}_{(i,j)}\right)\end{aligned} \tag{5.26}$$

Again, the previous frame is selected in the same way as was done for the P-frame. This is a very simple preprocessing method. However, it does require some intense processing. For example, to decide which previous frame to use for processing P and B-frames requires the error for the both possible previous frames to be calculated. The paper gives no indication of how to obtain the filter coefficients or of the processing requirements.

Zlokolica **[111]** presents a simple neighbourhood filter for removal of Gaussian noise. The system presented uses a 3 x 3 x 3 window, thus taking samples temporally as well as spatially. The algorithm works using a few simple steps. First the contents of the window are stored in a one dimensional array in any order as shown by equation 5.27.

$$Window = (x_1, x_2, \ldots, x_N) \tag{5.27}$$

where x_n is the nth sample in the window and N is the number of samples (27 for a 3x3x3 window). The elements of this array are then sorted according to their absolute difference to the centre pixel, $I(x, y, t)$, of the window:

$$Window_{sorted} = (x_{(1)}, x_{(2)}, \ldots, x_{(N)}) \tag{5.28}$$

where $x_{(1)} = I(x, y, t)$ is the centre pixel in the 3x3x3 window and $x_{(i)}$ are ordered such that $|x_{(1)} - x_{(i)}| < |x_{(1)} - x_{(j)}|$ for $i = 2 \ldots N$ and $j = i \ldots N$. The final output of the filter is computed using:

$$y = \frac{1}{\gamma} \sum_{i=1}^{N} \alpha_i x_{(i)} \tag{5.29}$$

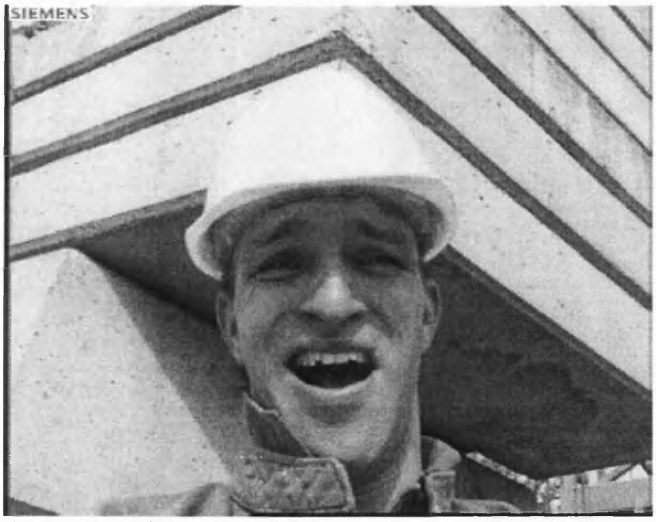
where α is a set of weights and γ is given by:

$$\gamma = \sum_{i=1}^{N} \alpha_i \tag{5.30}$$

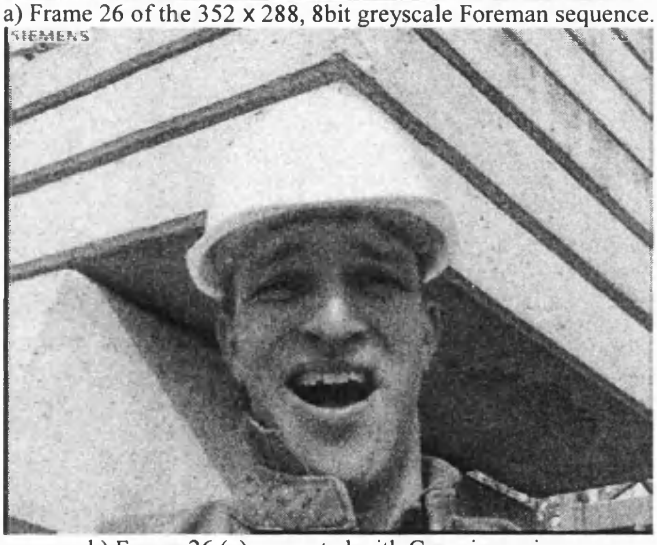
The implementation presented uses a simple weighting system. The number of neighbours to use is chosen, *M*, and the weights are then assigned using equation 5.31. For $M = 1$, no filtering is done at all, conversely, when $M = N$ the filter simply becomes a mean filter.

$$\alpha_i = \begin{cases} 1 & 1 \le i \le M \\ 0 & otherwise \end{cases} \tag{5.31}$$

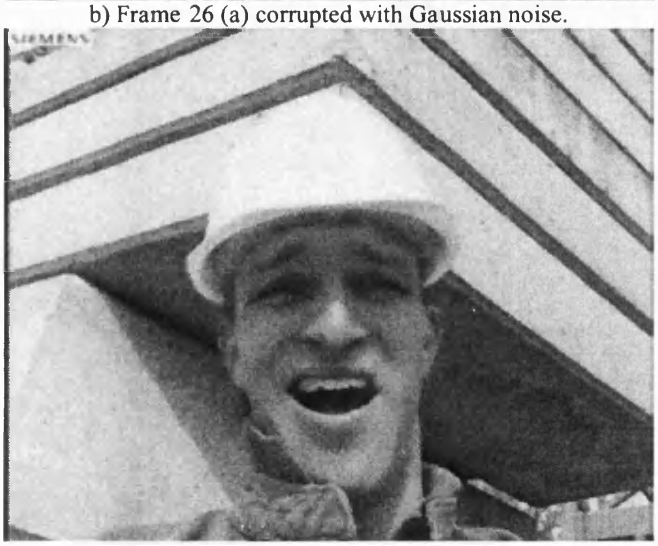
This is a simple filtering method, which can be implemented efficiently. However, the best value of *M* to use will depend upon the image content and the amount of noise present. Thus it is difficult to produce the optimum output for sequences where the original sequence is not available for reference. This method also processes every pixel in the sequence, which inevitably leads to blurring of the image, and degradation of image details as shown in Figure 5.12.



a) Frame 26 of the 352 x 288, 8bit greyscale Foreman sequence.



b) Frame 26 (a) corrupted with Gaussian noise.



c) Noisy image (b) filtered using the K-NN filter with $M=19$.

Figure 5.12: Example of using the K-NN filter.

There are still however several other methods that have been researched. Artificial Intelligence (AI) has been used extensively in various aspects of image processing including noise reduction in images. Haritopoulos et al present such a system for removing multiplicative noise using a Self Organising Map, namely Kohonen's **[112]**. Fuzzy logic, which is related to AI, has also been used for image filtering **[113]**, **[114]**. This uses a fuzzy logic control system to determine how to filter parts of the image. For example, if a region has an edge in it, then the filter will filter along the edge and not across it. Thus the edges are still filtered but not distorted. Also, the filters coefficients can be selected by using fuzzy logic to further optimise the filtering. This produces some good results but can be complex to implement and slow to run. This is because the location of the edges is required which means that some form of edge detection must be applied as well as someway of interpreting the edge detection.

Alternatively a combination of existing filtering techniques can be used [115]. This method uses a Laplacian filter to process parts of the image with little detail. The coefficients of the filter are altered when edge information is to be filtered in order to preserve the detail. However, this still requires some way of deciding if an area has any detail and if it is worth keeping. For use in image sequences, often the spatial techniques can be applied either as spatial only methods to each frame or they can be extended to incorporate the temporal correlation. There are numerous ways this can be accomplished, for example Motion-compensation may be used [116]. Tsuji presents a prefiltering system designed to improve the subjective quality of compressed video at low bit-rates, typically a few hundred-kilo bits per second [117]. This method removes noise but in addition also removes visually redundant structures, which significantly improves the compressibility of the video sequence. The method described is based on anisotropic diffusion [118] which preserves edges but allows control over which edges are kept and which are smoothed. A maximum of a 30% increase in compression is reported.

5.3.4 Mathematical Morphology for Noise Reduction

Several solutions to noise removal by the use of mathematical morphology have already been proposed [76], [54], [57]. Sivakumar and Goustasis use standard morphological operators with varying structuring elements to find the optimal filter for removing impulse noise from images [57]. However, one of the drawbacks to this method is that the structuring elements must be found that gives the optimum resultant image. Schonfeld and Goutsias [54] develop an optimal solution to the binary restoration problem. Essentially they find the best structuring elements to filter the image with by using the pattern spectrum. The noisy images are filtered using both the AF and ASF filters whereby several structuring elements of different sizes and shapes are used to form the optimal ASF filter. Mathematical morphology filtering techniques can also be used in conjunction with other methods such as the median. [119] present such a method called the Morphological Signal Adaptive Median (MSAM) filter. Binary dilations and erosions are used to control the window size, which is used by an impulse detector to locate the noise. The noise is then filtered by the modified median.

5.3.4.1 Morphological Image Cleaning

Peters describes a Morphological Image Cleaning (MIC) algorithm for reducing scanner and grain noise to improve the compression performance of the JPEG CODEC [120]. This method relies on the two basic characteristics of images used by many noise removal and image restoration techniques:

- Edges, thin lines and small features are clear and sharp,
- Areas between features are smoothly varying.

The basic principle of this system is to segment the image into features and noise. This is done in several stages. Firstly the image is smoothed using an Open-Close and a Close-Open to give a mean Open-Close-Close-Open (OCCO):

$$OCCO_{mean}(I,B)=\frac{(I\odot B)+(I\odot B)}{2} \tag{5.32}$$

where I is the image and B is the structuring element. Only disk shaped structuring elements are use. The structuring element has a changeable radius and can also be a FSE or non-FSE. The advantages and disadvantages of each method are discussed further in the paper. The image is filtered using several structuring elements with increasing size, which creates a morphological size distribution . Thus the original image is filtered using a sieve structure to form images with decreasing levels of detail. Hence the smoothed image at level J is given by:

$$OCCO_{mean}(I,B_J)=\frac{(I\odot B_J)+(I\odot B_J)}{2} \tag{5.33}$$

The method described uses the difference between levels, which they call residuals given by:

$$R_J = OCCO_{mean}(I,B_{J-1})-OCCO_{mean}(I,B_J) \tag{5.34}$$

where J is the level. R_J will by definition contain features of size J. Each residual, R_J, is assumed to contain both noise and features. Regions with a large amplitude in R_J are assumed to be of visual importance. The residual is a signed image and hence the light and dark regions are separated using equations 5.35 and 5.36 respectively. The features and noise are then separated using a simple threshold technique to generate a support map for the light and dark regions as given by equations 5.37 and 5.38 respectively.

$$L_J(x,y)=\begin{cases}R_J & \text{if } \mathrm{R}_J>0\\ 0 & \text{Otherwise}\end{cases} \tag{5.35}$$

$$D_J(x,y)=\begin{cases}-R_J & \text{if } \mathrm{R}_J<0\\ 0 & \text{Otherwise}\end{cases} \tag{5.36}$$

$$X_{L_J}(x,y)=\begin{cases}1 & \text{if } \mathrm{L}_J(x,y)\geq t\\ 0 & \text{Otherwise}\end{cases} \tag{5.37}$$

$$X_{D_J}(x,y)=\begin{cases}1 & \text{if } \mathrm{D}_J(x,y)\geq t\\ 0 & \text{Otherwise}\end{cases} \tag{5.38}$$

where t is the threshold value. The support maps will contain many isolated pixels of size 1 or 2, which are assumed to be noise. These pixels are then removed using a rank filter with a 3x3 mask, the result is then dilated and a logical *AND* is performed with the support map. The resultant support map should only contain active pixels that are near at least two other active pixels, but will sometimes contain isolated pixels. This operation is iterated until no isolated pixels remain. To recover the basis of the features, the non-zero regions of L_J and D_J are expanded by using their morphological skeletons and dilatations of their skeletons. Pixels that are not in a dilated skeleton are set to zero in the corresponding residual image (L_J or D_J). The clean image is then generated through combining the residuals with the original image. The coarsest image, which is $OCCO(I,B_J)$, is used as the base. Light features are put back in by adding to the sum of all of the cleaned light residuals (L_J) to the base. Dark features are put back in by subtracting the sum of all the cleaned dark residuals (D_J) to the base as given by:

$$I_{cleaned} = OCCO(I,B_k)+\sum_{J=1}^{k}L_J-\sum_{J=1}^{k}D_J \tag{5.39}$$

5.4 Preprocessing

Preprocessing is used to improve the video sequence for subsequent compression by removing high frequency information from the source material. Much of the information removed will be noise, unwanted data that has somehow been added to the image, but it is also possible to remove other information relating to image features, providing the perceived image quality is not lowered. In general there are two basic types of preprocessing systems:

- Those which attempt to reduce the noise present in the image,
- Those which attempt to simplify the image.

There are several ways in which a preprocessing system can be implemented:

- Integrated system,

  This type of system is usually integrated into a CODEC so that various information in the CODEC may be used to improve compression. For example, the quantisation can be adjusted according to the bit rate of the output of the CODEC, the motion compensation in the CODEC could be used in order to identify what should be filtered. In most cases this produces the most efficient forms of preprocessors in terms of computational efficiency.

- Standalone system,

  This system is not integrated into a CODEC and will process the video independently of the CODEC. However, this system could be designed in such a way that it could be used for and CODEC, or it may be design for use with only one specific CODEC.

Both of these methods have their own distinct properties. For example, having an integrated system may help to reduce the workload of the preprocessing, but it also ties the processing to be of use to a few specific CODEC's. However, a standalone system could be used on with any CODEC. This may result in a slight loss in computational efficiency, but does mean that it does not need to be rewritten to fit into new CODEC's.

There are several definitions of image simplification, which depend on the application for which they are intended to be used with. For example, for image segmentation, it maybe advantageous to simplify the image by removing noise and small objects prior to segmentation to increase the efficiency of the segmentation algorithm. Several methods exist to in order to accomplish image simplification. In particular, [53] presents a segmentation system that uses AM (see section 4.6.1) to simplify the images before segmentation. This method uses an AO and AC to a fixed area size of 100. The regions can then be pulled out and segmented. However, noise removal or reduction can also be classified as image simplification as it simplifies the quantisation stage of the CODEC resulting in a more compressible image.

5.5 Conclusion

Most preprocessing methods have been designed for impulse type noise, as it is relatively simple to filter using the median filter and its derivatives. This is because the median always uses one of the input values as the output value. This limits the usefulness of median filters to impulse noise removal although several attempts at using them for Gaussian noise removal have been made [104]. In conclusion, median filters appear advantageous as they produce good results, are easy to implement and fast to run. However, Gaussian noise is found more often in the real world as shown in [121] which attempts to remove scratches and photographic grain noise from archive film material. Most of these systems use some form of median filter. Morphological filters can also be classified as order statistics filters and hence have similar properties to the median such as edge preservation [122],

[123]. Median filters have been researched in great detail and are widely used, whereas morphological filters have not been researched in great detail, especially for use in preprocessing. Due to these points, a fresh view has been taken using morphological filters. Morphology is looked at in detail in chapter **4**. It is clear from the overview this chapter has given on preprocessing, that most current methods are specifically set-up to work with one CODEC and only one type of noise. This limits the usefulness of preprocessing systems. Hence the preprocessing system developed within this thesis is to be CODEC independent so that will work with several different systems.

6 Image Preprocessing using Attribute Morphology

Chapter 2 showed what digital video is and the advantages to be gained from preprocessing. Current methods for preprocessing video are often complex and have been extensively researched [54], [57], [76], [112] - [117], [119], [120]. However, chapter 4 showed that mathematical morphology is a relatively simple and effective non-linear class of techniques that can have application to preprocessing digital images and video. In addition there has been no in-depth investigation done into using mathematical morphology for preprocessing. For these reasons and the fact that it is edge preserving, mathematical morphology has been chosen as the area of investigation for this research. Whilst the basic morphological operations (dilation, erosion, opening, etc.) simplify an image, the results are not visually acceptable, especially for use in digital television. For this reason, attribute morphology has been investigated and developed upon.

This chapter includes measured timings. Each of these timings is run over at least 50 images so that any inconsistencies are minimized and more realistic results are obtained. The system used to run the timing tests contains an AMD Athlon XP 1800+ CPU, running at 1.53GHz, 512MB PC2700 DDR RAM and running Windows XP Home Edition operating system. All timings reported throughout this document were obtained using this system so that all results are directly comparable. The Microsoft Visual C++ Version 6 programming environment was used to program all of the source code contained within this thesis [124]. This then allows a direct comparison between the speed of the code. Also, where possible the Standard Template Library (STL) has been used to increase performance [125]. In addition this chapter describes how attribute morphology has been implemented and improved on, specifically for the task of preprocessing digital video sequences for simplification and noise removal. There are two main areas of interest. The first is for noise removal and the second is for image simplification, where the idea is to filter the video in such a way that the images are simplified but are visually imperceptible to the viewer. The output of either of these filters can then be applied to any CODEC.

Section 4.5 defined AM. Equations 4.31 and 4.32 are not very efficient and of little use due to the fact that they essentially try every possible combination of openings. A better method can be created by using maximum and minimum points inside the image as all other points are unaffected by the opening and closing operations. The method described below is just one interpretation of the above, of which there are various different interpretations in use [77]. This technique does not use a structuring element as before. Instead it adapts its shape to the image. First the maximum points of the image are found, both local and regional .

The maximum points are extracted and labelled as regions after which each region is processed individually (see Figure 6.2). If a region has an area equal to 1, then the neighbours are inspected (see section 6.1.2) and the maximum value is found. If the maximum value is smaller than or equal to the current value of the region, then the neighbour is added to the region and the value of the region is changed to that of the neighbour added. This is then repeated for regions of area size 2 provided that they are still maximas. The area size is incremented by 1 every time until a predetermined target area size is reached. When the above process reaches this target or if all the regions become inactive, then the process is stopped. In addition, the above method can also be applied in 3D (or higher) by looking at the neighbours on the previous and next frames. An AC (see section 4.6.1) is processed in a similar way. It starts by using the minimum image points and instead of using the maximum value of the neighbours; it uses the minimum value so long as it is equal to or greater than the current region value. Figure 6.3 shows an image filtered with the area-open method. It can clearly be seen that this is an improvement over the standard opening method in that more of the image detail is retained, even with a high degree of filtering. Three different implementations are now looked at and the performance of each is assessed.

Local maximum is a pixel that is larger that all of its neighbours. Regional maximum is a region/group of pixels that are larger than all of their neighbours.

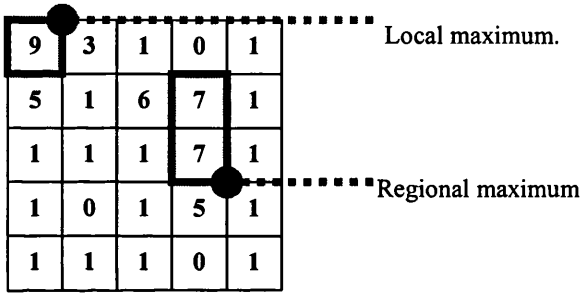
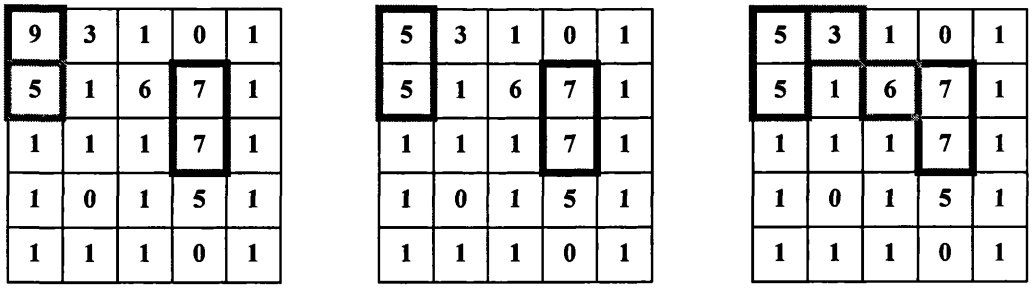


Figure 6.1: Example to show local and regional maximas.



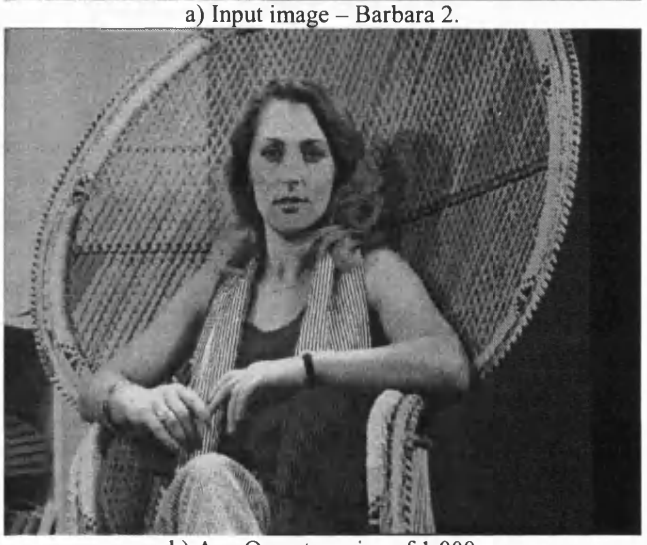
For regions of size 1, find the maximum neighbour that is smaller than the regions value.

Add the pixel to the current region by changing the regions value to the new pixel.

Repeat for regions of size 2.

Figure 6.2: Example of how Area-Opening works.

a) Input image – Barbara 2.



b) AreaOpen to a size of 1,000

c) AreaOpen to a size of 100,000

Figure 6.3: A greyscale Area-Opening on the 720 x 576, 8bit greyscale Barbara 2 image.

6.1 Pixel Queue Algorithm

AM (see section 4.6.1) was initially introduced by Cheng and Venetsanopoulos, which they called NOP and NCP operators [76]. Vincent was the first to propose a system, which allowed a practical method in which area morphology could be implemented [78]. The algorithm for an AO (see section 4.6.1), operating on an image (*I*), using Vincent's method, which is commonly referred to as the Pixel Queue, is shown in Listing 6.1. To implement the pixel queue method, the extrema regions within the image need to be located. This can be done in several ways. The most widely adopted method uses a technique called '**morphological reconstruction**' [77], [78], [126] - [128].

1. Extract the regional maxima (*m*) of *I* [77],
2. For each *m* of *I*, do the following:
    a) If the area of this maxima is larger that the target size (λ), then move onto the next maxima,
    b) Find the highest greyscale valued neighbour (*n*) of *m*,
    c) If the value of *n* is greater than the value of the maxima, then *m* is no longer a maxima so move onto the next maxima (go to step 2),
    d) Add pixel *n* to the maxima *m*,
    e) Set the value of all pixels belonging to *m* to be the value of *n*,
    f) Repeat this procedure, from step a until area(*m*) is greater than λ.
3. Repeat step 2 until all *m* have been processed.

Listing 6.1: Vincent's Pixel Queue Algorithm

6.1.1 Morphological Reconstruction

Morphological reconstruction extracts the connected components of an image using markers from another image [77], [78], [126] - [128]. This is done by using Geodesic Distance, although there are other ways in which to implement reconstruction [77]. For example level sets (see section 4.2) could be used. However level sets are a less efficient method due to the way it is required to be implemented, which is supported by work done by others [77]. Geodesic distance is simply the shortest distance connecting two points inside the same set as shown in Figure 6.4. Note that the length will actually depend on the type of connectivity used. As stated above, in order to perform reconstruction, the geodesic distance is required. This is done by using a method called Geodesic Dilation which is defined as:

$$\delta_I^{(n)}(J) = \{p \in I : d_I(p, J) \leq n\} \tag{6.1}$$

where J is the marker image and I is the mask image, $d_X(p,Y)$ is the distance induced by I between p and J. This basically finds connected points within a given size. From this equation, it can be said that the Geodesic dilation of size n can be obtained by iterating n elementary Geodesic dilations as given by equation 6.2.

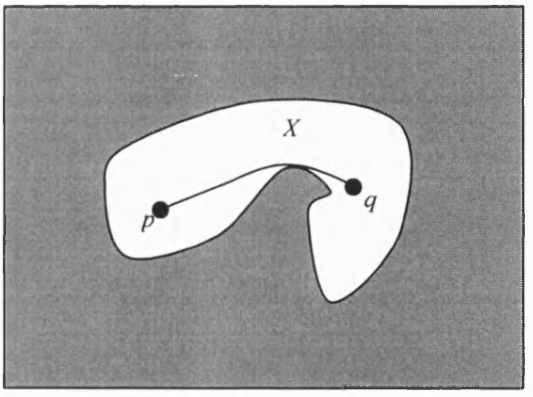


Figure 6.4: Geodesic distance between points *p* and *q* within set *X*.

$$\delta_I^{(n)}(J) = \delta_I^{(1)}\left(\delta_I^{(1)}\left(\delta_I^{(1)}(\cdots(J))\right)\right) \tag{6.2}$$

where the elementary Geodesic dilation for size 1 is defined as:

$$\delta_I^{(1)}(J) = (J \oplus B) \cap I \tag{6.3}$$

where J is the marker image and I is the mask image. Note that this equation is only valid for elementary geodesic dilations. This works by dilating the marker image and taking the intersection between the result and the mask image. The result finds pixels that are a distance of 1 away from the pixel being operated on. By applying successive geodesic dilations of the marker image (J), inside the mask image (I), connected components of I which intersect with J are progressively lowered in value. Using this knowledge, reconstruction can be defined as:

$$\rho_I(J) = \bigcup_{n \geq 1} \delta_I^{(n)}(J) \tag{6.4}$$

This iterates the dilations until they become stable (i.e. values do not change any further). This will work only on peaks (i.e. ones). This will become clearer in the next chapter. To do the opposite, the dual reconstruction is used. This uses Geodesic erosion, which does exactly the same as dilation, except on zeros (i.e. finds connected components of the zeros). Geodesic erosion is defined by equation 6.5 and the elementary geodesic erosion that it uses is given by equation 6.6.

$$\varepsilon_I^{(\ )}(J) = \varepsilon_I^{(1)}\left(\varepsilon_I^{(1)}\left(\varepsilon_I^{(1)}(\cdots(J))\right)\right) \tag{6.5}$$

$$\varepsilon_I^{(1)}(J) = (J \ominus B) \cup I \tag{6.6}$$

Hence dual reconstruction can now be defined as:

$$\rho_I^*(J) = \bigcap_{n\geq 1} \varepsilon_I^{(n)}(J) \tag{6.7}$$

6.1.1.1 Extending to Greyscale

From chapter **4**, it is clear that 6.3 can easily be redefined in grey-scale as:

$$\delta_I^{(1)}(J) = \left(\max_{m\in B}(J_{x+m})\right) \wedge I = (J \oplus B) \wedge I \tag{6.8}$$

where $\wedge$ is the point-wise minimum and B is a FSE (see section 4.2). Note that to dilate to size n, equation 6.2 is still used, but with the above definition for the elementary Geodesic dilation. Reconstruction can thus be redefined as:

$$\rho_I(J) = \bigvee_{n\geq 1} \delta_I^{(n)}(J) \tag{6.9}$$

This is repeated until it becomes stable (i.e. does not change any further). Similarly, geodesic erosion and dual reconstruction can be redefined as:

$$\varepsilon_I^{(1)}(J) = \left(\min_{m\in B}(J_{x+m})\right) \vee I = (J \ominus B) \vee I \tag{6.10}$$

$$\rho_I^*(J) = \bigwedge_{n\geq 1} \varepsilon_I^{(n)}(J) \tag{6.11}$$

where $\wedge$ is the point-wise minimum, $\vee$ is the point-wise maximum and B is a FSE. Again, this is repeated until it is stable. Equation 5.8 is still valid, but uses the above definition for geodesic erosion. The mask is usually set to be the signal. However there is no indication of what should be used for the marker. If the original signal is used, but with 1 subtracted from every element, the maximum regions are lowered in value by one. The result is then subtracted from the original input, the maximum regions are marked as shown in Figure 6.5. Thus reconstruction can now be redefined as a marker to locate maximum points as:

$$M^{\max}(I) = I - \rho_I(I-1) \tag{6.12}$$

An interesting and useful property of this, is that by subtracting different values from *I*, for example, regions can be extracted that fluctuate. This is known as '**Dome Extraction**' or the '**h-dome**' and is given by equation 6.13, which is shown more clearly in Figures 6.6 and 6.7. This is called dome extraction for the reason that it extracts the top or bottom parts of a signal (i.e. the peaks and/or troughs). Although Granules (see section 4.5) do the same, they are usually referred to as the information that falls through a sieve as the data is progressively filtered more. Hence for the remainder of the thesis, Granules specifically refer to the sieve structure and dome extraction to extracting the extrema, that is the h-domes, which is the general concept used by many authors [77]. To find the minimum points, a similar method is used as is given by equation 6.14. This works in a similar method, but extract minimum regions rather than maximum regions.

$$M_h^{\max}(I) = I - \rho_I(I-h) \tag{6.13}$$

$$M_h^{\min}(I) = \rho_I^*(I+h) - I \tag{6.14}$$

1	3	7	5	1	2	5	5	1	Input signal (*I*)
0	2	6	4	0	1	4	4	0	Marker signal *J* = *I-1*
1	1	1							Structuring element
2	6	6	6	4	4	4	4	4	$J \oplus B$
1	3	6	5	1	2	4	4	1	Minimum between *I* and $J \oplus B$

The output is not the same as the input, so repeat, using the result as *J*.

3	6	6	6	5	4	4	4	4	$J \oplus B$
1	3	6	5	1	2	4	4	1	Minimum between *I* and $J \oplus B$

The output is the same as the input so stop. To get the regions marked, subtract the result from I as shown below.

0	0	1	0	0	0	1	1	0

Figure 6.5: Maxima extraction.

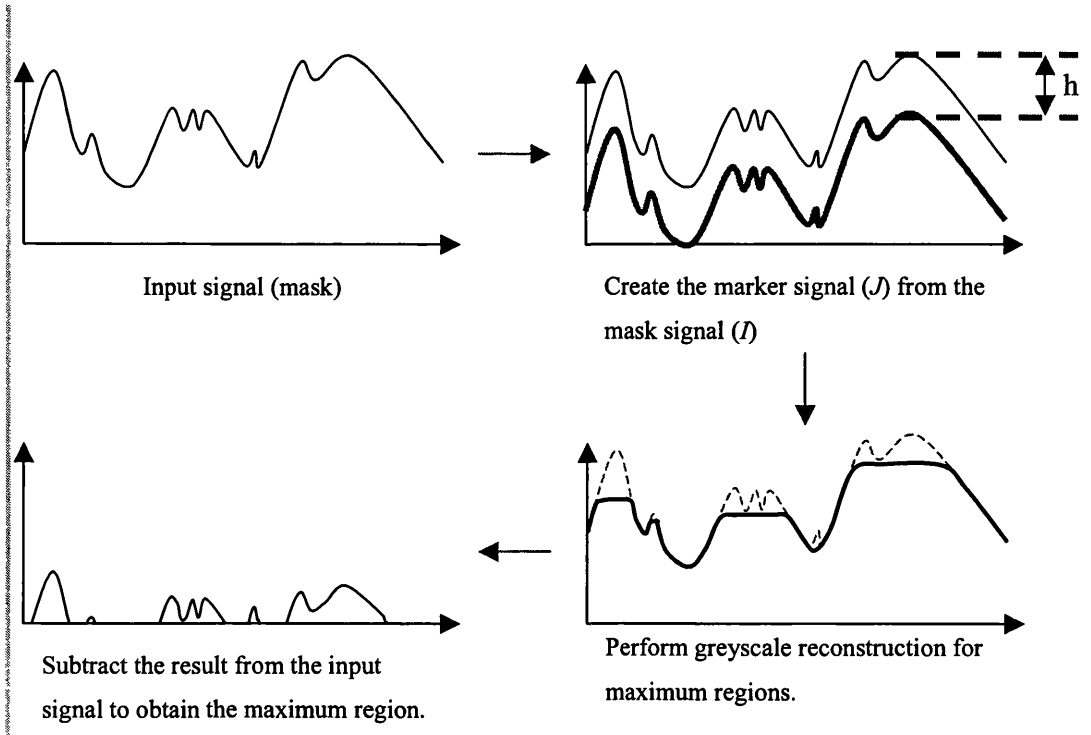


Figure 6.6: Dome extraction.

1	3	7	5	1	4	5	5	3	Input signal
0	0	1	0	0	1	1	1	0	Marked regions with $h=2$
0	0	1	1	0	1	1	1	1	Marked regions with $h=3$

Figure 6.7: Maxima extraction with fluctuations.

6.1.1.2 Extending to 2D

To use this technique on images, the elementary dilation and erosion are redefined as given by equations 6.15 and 6.16 respectively. In addition, the structuring element is changed to 2D in the same way as shown in chapter 4.3. This can also be changed to work in 3D in exactly the same way (see section 4.4). Assuming that the image contains noise, then by using different values of h, fluctuations caused by noise can be removed.

$$\delta_I^{(1)}(J) = \left(\max_{(m,n)\in B} \left(J_{x+m,y+n} \right) \right) \wedge I = (J \oplus B) \wedge I \tag{6.15}$$

$$\varepsilon_I^{(1)}(J) = \left(\min_{(m,n)\in B} \left(J_{x+m,y+n} \right) \right) \vee I = (J \ominus B) \vee I \tag{6.16}$$

This method can be implemented easily using basic morphological operations and the more simplified algorithm given in Listing 6.2. However, there are some properties of the reconstruction process, when used for extrema extraction, that can be exploited to increase the efficiency of the implementation as can be seen from Tables 6.1 and 6.2. Table 6.1 also shows that this method can take a significant amount of iterations, up to 26 for the images test, to stabilise and up to 600mS. Several more efficient methods [77] have been devised as discussed in the following sections.

1. Let I be a mask image (this may be binary or greyscale)
2. Let J be a marker image, defined on the domain D_I, $J \leq I$,
3. Allocate memory for temporary work image, K, defined on D_I,
4. Repeat until stable:
   a. **Dilation step;**
      For every pixel $p \in D_I$, $K(p) \leftarrow \max\{J(q), q \in N_C(p) \cup \{p\}\}$
   b. **Point wise minimum;**
      For every pixel $p \in D$, $J(p) \leftarrow \min(K(p), I(p))$

Listing 6.2: Standard morphological reconstruction.

Connectivity	4nn		8nn	
Reconstruction method	Standard Reconstruction	Sequential Reconstruction	Standard Reconstruction	Sequential Reconstruction
Barbara	18	**4**	15	**4**
Barbara 2	26	**4**	26	**4**
Boats	19	**4**	16	**4**
Goldhill	14	**4**	14	**4**

Table 6.1: Number of complete scans to reconstruct the image. Shaded cells show best results.

Connectivity	4nn			8nn		
Reconstruction method	Standard	Sequential	Hybrid	Standard	Sequential	Hybrid
Average Time (mS)	346	132	**30**	401	154	**45**
Times faster than standard method	1.00	2.61	**11.53**	1.00	2.60	**8.91**

Table 6.2: Average execution times (mS) for the different reconstruction processes taken over 76 8bit greyscale image of size 720 x 576 pixels. Shaded cells show the best results.

6.1.1.3 Sequential Reconstruction

By scanning an image in a predefined order, for example raster or anti-raster, more efficient algorithms can be used. This also allows the new value of the current pixel to be written directly to the image being worked on so that it can be used by then next unprocessed pixel also decreasing the memory requirements. First the structuring element is split into two parts as shown in Figure 6.8. The reconstruction algorithm is then split into a simple algorithm, using the Mask (I), Marker ($J, J \leq I$) and connectivity C:

1. Let I be a mask image (this may be binary or greyscale)
2. Let J be a marker image, defined on the domain D , $J \leq I$,
3. Repeat until stable:
   a. Scan J in raster order:
      i. Let p be the current pixel,
      ii. Update p by:

$$J(p) \leftarrow \left(\max\left\{ J(q), q \in N_C^+(p) \cup \{p\} \right\} \right) \wedge I(p)$$

   b. Scan J in anti-raster order:
      i. Let p be the current pixel,
      ii. Update p by:

$$J(p) \leftarrow \left(\max\left\{ J(q), q \in N_C^-(p) \cup \{p\} \right\} \right) \wedge I(p)$$

Listing 6.3: Sequential reconstruction.

This method results in significantly less image scans, for the four test images used, the entire scans needed for the standard method and this method is shown in Table 6.1. It can be seen that this is a dramatic increase in performance when compared to the standard method. Also it should be noted that this method will not always takes 4 complete scans to complete. The actual increase in performance can be seen from Table 6.2. This method is on average 2.7 and 3.14 times faster than the standard method for 8nn and 4nn respectively.

6.1.1.4 FIFO Reconstruction

The previous two implementations have worked upon processing all of the pixels within the image. However only a few pixels will actually be changed in the final output. This can be used to create a more efficient implementation by only processing pixels that will be modified. This implementation works in two parts. Firstly those pixels that will be modified are identified, which are the boundaries of maximas within the marker image. In effect, the marker image that is actually used consists only of the regional maxima as given in equation 6.17. These are added to a First In, First Out (FIFO) queue. The second step propagates the pixels in the FIFO. The algorithm for this process is given in Listing 6.4. This algorithm is specifically designed for the reconstruction process, as it requires the maxima of the marker image in order to work. However, it is the maximas that are required to be extracted. Hence this method is of little use.

$$\forall p \in D_I, \quad R(Marker)(p) = \begin{cases} I(p) & \text{if p belongs to a maximum,} \\ 0 & \text{Otherwise} \end{cases} \tag{6.17}$$

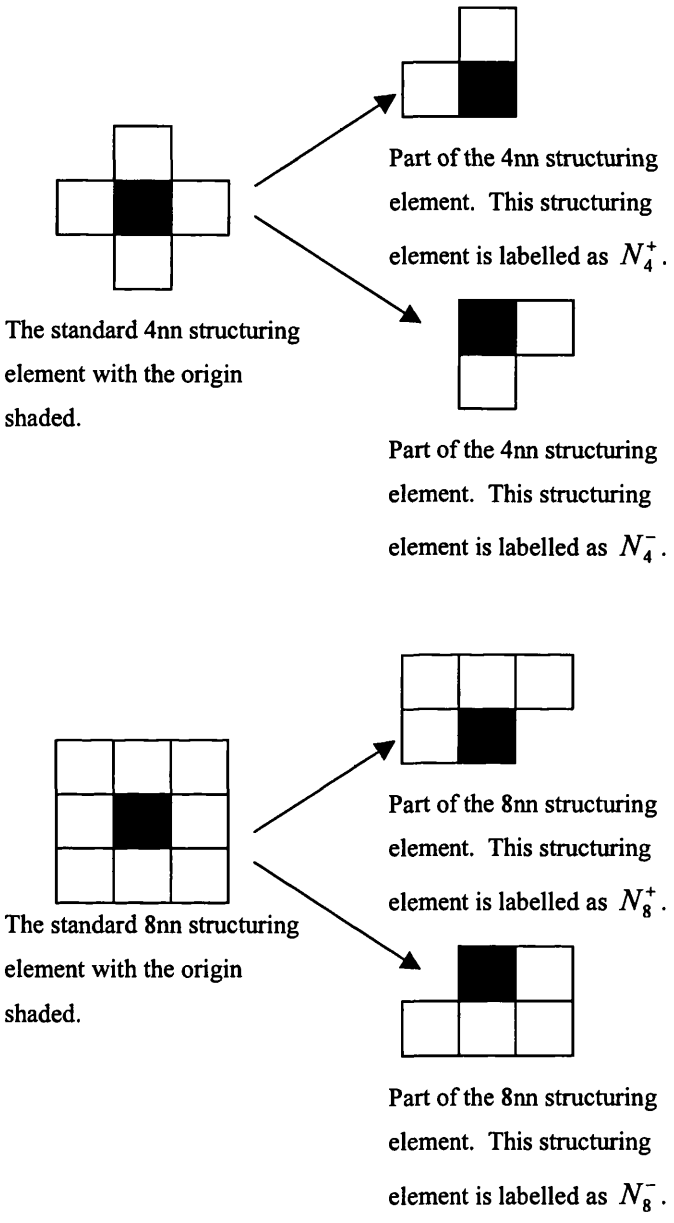


Figure 6.8: How to decompose the structuring element for efficient implementation of reconstruction.

1. Let I be a mask image (greyscale)
2. Let J be a marker image, defined on the domain D_I, $J \leq I$,
3. Compute the regional maxima of J: $J \leftarrow R(J)$,
4. Initialise the FIFO queue with the boundaries of the maxima:
   a. For every pixel $p \in D_I$:

      Add p to the FIFO *iff* $J(p) \neq 0$ AND $\exists q \in N_G(p), J(q) = 0$
5. Propagation:
   a. Until the FIFO is empty, do:

      $p \leftarrow$ first element in the fifo,

      For every pixel $p \in D$:

         If q is lower than p and if it is necessary to propagate it:

            If $J(q) < J(p)$ and $I(q) \neq J(q)$:

               $J(q) \leftarrow \min\{J(p), I(q)\}$

               add q to the FIFO.

Listing 6.4: FIFO reconstruction.

6.1.1.5 Hybrid Reconstruction

The previous method has the drawback that it needs to know where the regional maxima are located in order to perform the reconstruction. The maxima can easily be obtained using one of the earlier methods. However this defeats the point, as it is the maximas that are required. The hybrid method merges the ideas of the sequential and FIFO methods. The image is raster scanned first and then anti-raster scanned adding pixels onto a queue. The queue is then cycled through changing and adding pixels until it is emptied. The algorithm for this method is given in Listing 6.5 and only performs two complete image scans initially to fill the FIFO with its starting points. Thereafter, only pixels that are to be modified are touched significantly decreasing the amount of processing required. To show the efficiency of this method compared to the standard and sequential methods, the average time to execute was taken for 76 8bit greyscale images with a size of 720 x 576, the results of which are given in Table 6.2. This method is on average 11.53 and 8.91 times faster than the standard method for 4nn and 8nn respectively. In addition, the 4nn connectivity proves to be faster to reconstruct than the corresponding 8nn. For example, the 4nn sequential method is 1.17 times faster than the 8nn sequential method, thus providing an advantage for using 4nn compared to 8nn. This is shown further by evaluating the average execution time to reconstruct images with the value of h set to 1 (see Figure 6.9). Each image size has a minimum of 50 8bit greyscale images. The results show again that the hybrid method is best, which is shown to be more of an improvement with larger images.

1. Initialise the FIFO queue,
   a. Scan J in raster order:
      - Let p be the current pixel:

$$J(p) \leftarrow \left(\max\left\{ J(q), q \in N_C^+(p) \cup \{p\} \right\} \right) \wedge I(p)$$

   b. Scan J in anti-raster order:
      - Let p be the current pixel:

$$J(p) \leftarrow \left(\max\left\{ J(q), q \in N_C^-(p) \cup \{p\} \right\} \right) \wedge I(p)$$

      - If there exists $q \in N_C^-(p)$ such that $J(q) < J(p)$ and $J(q) < I(q)$ then add p to the FIFO.
2. Propagation step:
   a. While FIFO is not empty do:
      - $p \leftarrow$ *first element in the fifo* ,
      - for every pixel $q \in N_C(p)$:

        - If $J(q) < J(p)$ and $I(q) \neq J(q)$:
          - $J(q) \leftarrow \min\{J(p), I(q)\}$
          - add q to the FIFO.

Listing 6.5: hybrid reconstruction.

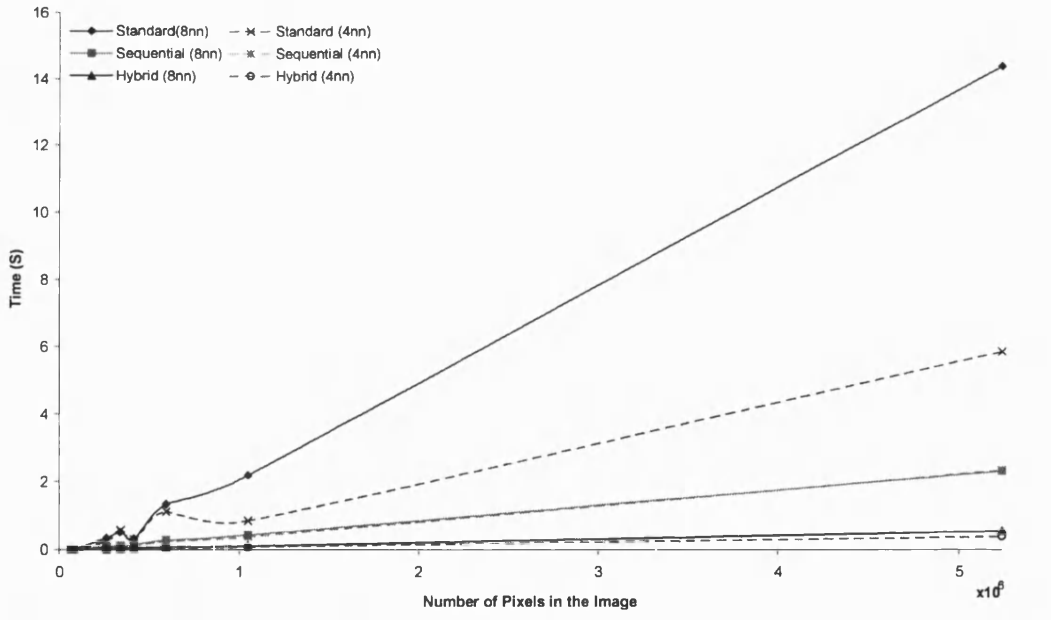


Figure 6.9: Average execution time against number of pixels in the image for 50 images of each size.

The hybrid reconstruction has been used as the basis for locating minimum and maximum regions for this thesis for both its speed and ease of implementation. Figure 6.10 shows the regions identified by using this method with various values of h. However, this method cannot find both the minimum and maximum points simultaneously. The implemented method finds the maximum points and then the minimum points sequentially. This has the disadvantage that some parts of the image may be defined as both minimum and maximum regions when h is larger than 1.

6.1.2 Region Growing

Step b of Listing 6.1 needs to recursively scan the neighbours of the region being processed. The most obvious way to implement this is to search all of the neighbours at each step. Whilst this saves computer memory, it is very inefficient in terms of computation speed. Thus the neighbours should be stored in memory. Using a simple unsorted array to store the neighbours is the simplest method. New neighbours can just be added at the end of the array. To find the neighbour to add to the region, the array is searched for the largest value, assuming the current region is a maxima. Thus if there are n neighbours, then to find the maximum neighbour to use will require n comparisons. Although this is more efficient than keep rescanning the neighbours, there are other more efficient methods. The next step is to add a sorting algorithm to the list so that the maximum neighbour is always the first element in the array. There are several sorting algorithms available to accomplish this with the best being the quick sort algorithm [130], [131]. The quick sort algorithm has an average sort time of $n \log(n)$ and a worst case of n^2 operations. However, the quick sort is faster than a simple exhaustive searching when $n<10$. Also there is the problem that whenever a new neighbour is added, the entire list is required to be resorted thus decreasing performance further.

6.1.2.1 Hierarchical Queues

Hierarchical Queues could also be used to store the neighbours **[78]**. A hierarchical queue is simply a stack of queues. For example, suppose that an image has 16 levels, then the corresponding hierarchical queue would have 16 levels. The neighbours of a region would be added to the queue at the level corresponding to its value. Thus when the region is grown, the hierarchical queue is simply scanned from the highest level down and the first non-empty level is the value of the highest neighbour. The actual neighbour can then be popped from the queue at that level. For the worst-case, an image with n levels then n operations would be required to find the next neighbour. Note that this is not taking into account the operations required by the queues themselves. Also this method has the drawback of requiring large amounts of memory. For example, consider an 8bit image, then there would be 256 levels and therefore 256 queues. Using the studio standard of 10bits, this increases to 1024 queues. Thus the memory requirements increase rapidly and therefore is not very memory efficient.

Input image – Barbara 2.

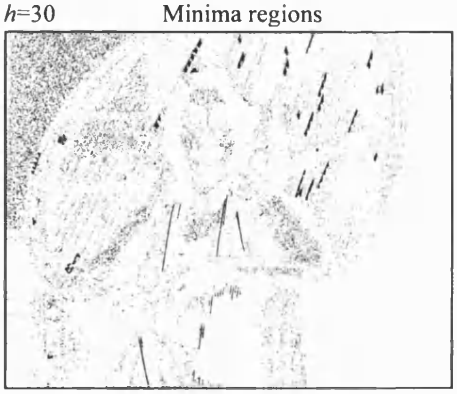
Maxima regions

Regions at h=1 Minima regions

Maxima regions

Regions at h=30 Minima regions

Maxima regions



Regions at h=60 Minima regions

Figure 6.10: Extracted regions using reconstruction on the 720 x 576, 8bit greyscale Barbara 2 image.

6.1.2.2 Priority Queues

Although for small values of n, the quick sort method would increase performance, another method is required that will work for a much larger number of neighbours and also require little work when a new neighbour is added or removed from the queue. Vincent proposed the use of Priority Queues [78]. A priority queue will store objects according to some predefined priority with the highest priority object being placed at the top of the queue. Thus the queue can be designed to store neighbours in order of their value. For example, storing neighbours for a maximum region, the priority would be that the highest values get placed at the top of the queue.

When the region is being grown, the top node will be pulled from the queue and added to the region. This saves searching through all of the neighbours and is there for a fast and efficient method requiring only 1 operation to extract the next neighbour. The neighbours of the element added to the region are then inserted into the queue. The queue can be thought of as a binary tree where each node in the queue can only have two child nodes as shown in . This has the advantage that it can be implemented using an array very efficiently. To find the index (element) of a parent or child of a given node, the following equations are used:

$$\text{Parent of a node} = \frac{\text{index of current node}}{2} \tag{6.18}$$

$$\text{Left child of a node} = 2 \times \text{index of current node} \tag{6.19}$$

$$\text{Right child of a node} = (2 \times \text{index of current node}) + 1 \tag{6.20}$$

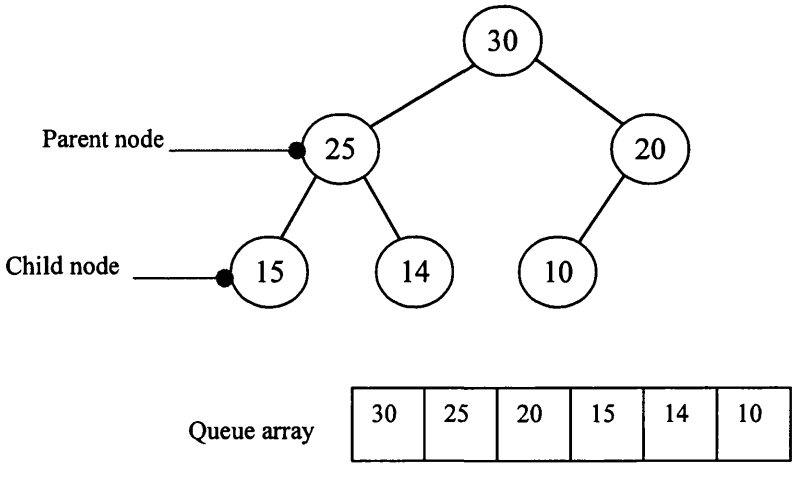


Figure 6.11: A priority queue.

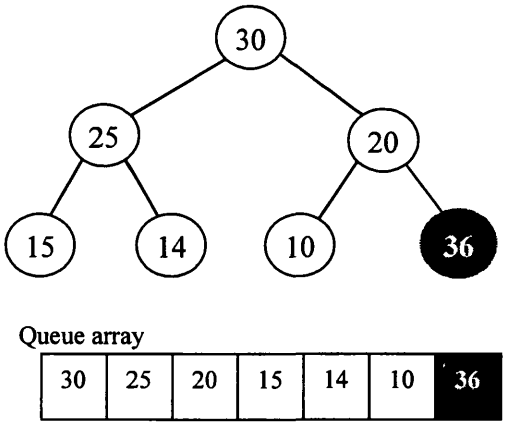
If a region is being grown, then new neighbour(s) will be added to the queue as shown in Figure 6.12. When the top node is removed (i.e. to be added to a region), then the queue must be sorted to restore the priority (see Figure 6.13). Both insertion and removal of a node can be implemented using $\log(n)$ operations [78], which is significantly faster than the quick sort algorithm. Obviously the initial set-up would require several insertions, but once growing commences, there should be only a few insertions at each iteration.

Priority queues provide an extremely efficient way, both in terms of memory and speed, in which to store the neighbours of a region. This has shown the basic principles behind the priority queue. Although there are other operations that can be applied to the queue, such as a resort or heap fix, the operations described here are all that is need to be of use for applying it for use in area morphology. The method described here has been implemented and used for this thesis. It has proved to be very efficient and useful for growing regions.

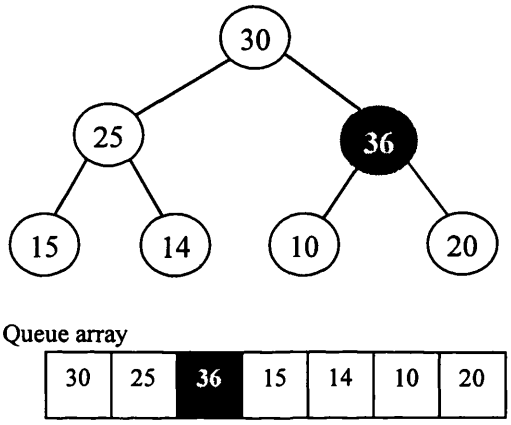
6.1.3 Performance of the Pixel Queue Method

Now that a method has been developed for extracting the regions and storing the neighbours the remainder of the algorithm is implemented as it is described in Listing 6.1. The efficiency of this system is then analysed over a range of area sizes from 1 to 10000 for both 4nn and 8nn on 50 images including a selection of several ITU test images (Barbara, Barbara 2, Boats and Goldhill). Figure 6.14, shows the average time taken to process an image. Using this method, the time take to process an image is dependent upon the image content, the area size and the connectivity used. The 4nn is seen to be faster to process than the 8nn. Similarly, the larger the area size, the longer it takes to process.

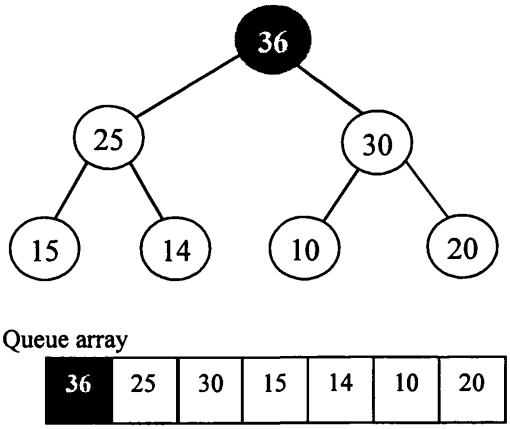
Meijster and Wilkinson have conducted similar timing experiments [132]. They show their own implementation of the pixel queue method executing an area size of 10000 in approximately 160mS, which is significantly faster. In addition, timing results are also given for the Max-tree and Wilkinson's implementation using Tarjan's union find method. However, the image dimensions and machine details are not given and hence a direct comparison cannot be made. Acton proposed alternative fast implementations to produce an approximation of the AO (see section 4.6.1) algorithm [133]. Timings are given for the pixel queue implementation using the Cameraman test image and a range of image sizes. For an image size of 128 x 128 the algorithm takes 102s, for 256 x 256 it takes 1437.3s and for 512 x 512 it takes 6046.6s. The area size used is not given, but using the maximum of 65536 (256×256), the method implemented for this thesis takes only 11.3s using the 256 x 256 cameraman image. However, the programming language and the connectivity used are not reported, which may explain why the results take considerably longer than the implementation written for this thesis. In addition, timing results are only given for the Cameraman image.



a) A new node (**shaded**) is added by placing it in the bottom of the queue (first empty element). This disrupts the priority of the queue and hence needs to be restored.

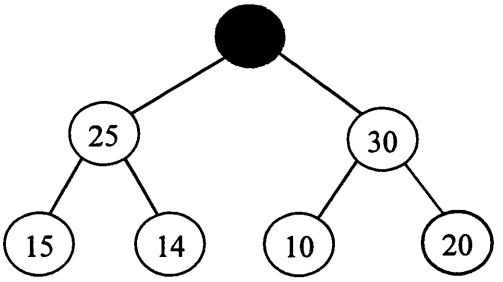


b) Percolate the new node up through the tree. This is done by comparing the node with its parent. If it has a higher priority, then swap the nodes and continue this action until the parent has a higher priority. In this case, 36 is larger than 20 so swap the nodes.



c) The new node is again larger than its parent so swap them. This is repeated until the node has a lower priority than its parent.

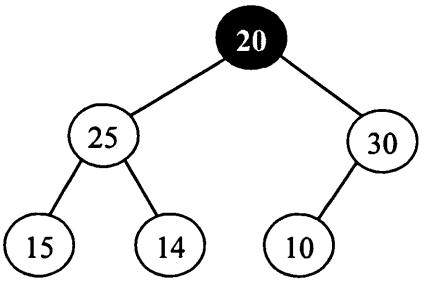
Figure 6.12: Example of how a new node can be added to the queue.



Queue array

	25	30	15	14	10	20

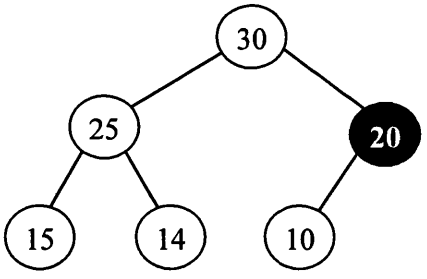
a) The top most node (**shaded**) has been removed. This disrupts the order of the queue and hence must be sorted to restore the queue properties.



Queue array

20	25	30	15	14	10	

b) To restore the queue properties, the bottom most node (the last element, **shaded**) is move to the top of the queue. This is then percolated down through the queue.



Queue array

30	25	**20**	15	14	10	

c) If any of the child nodes have a higher priority, then the node is swapped with the child that has the highest priority as shown. This is repeated until the node has a higher priority than its children.

Figure 6.13: Removal of a node from the queue.

6.1.4 Max-tree

Salembier introduced the idea that anti-extensive connected set operators can be represented by a tree structure [134]. For binary images, this is relatively straight forward as shown in Figure 6.15. Each component of the image is connected to the background of the image, which is also called the root. The tree is then filtered by simply visiting each node and evaluating it against an attribute. If the node fails to meet this criteria, then it is removed.

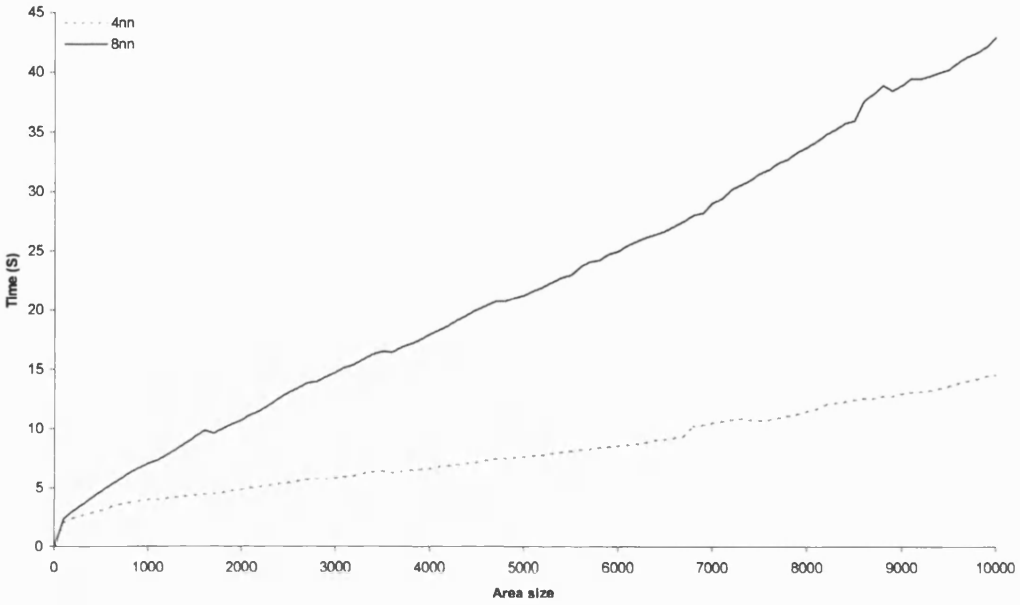
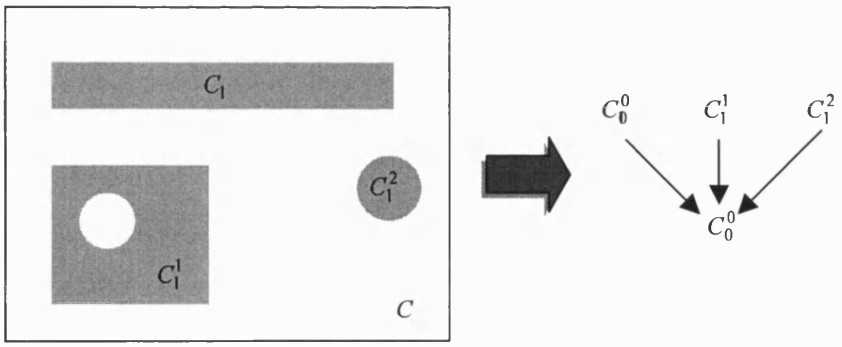


Figure 6.14: Timing results for the pixel queue method using 50 8bit greyscale images with a size of 720 x 576 pixels. Area size was measured at intervals of 100.



Binary image with the components labelled as C_h^k where h is the level and k is the node number.

The max-tree is created by connecting all components of $h=1$ to the root value, $h=0$.

Figure 6.15: A binary Max-tree.

The extension of connected components to greyscale uses a partition of flat zones. Thus threshold decomposition is used to create the max-tree as shown in Figure 6.16. The root node is first created by thresholding the image using the minimum value (h_{min}) of the image. All pixels equal to the threshold are placed in the root node, $C^0_{h_{min}} = \{H\}$. All pixels greater than the threshold are put into temporary nodes, $T^0_1 = \{A, B, C, D, E, F\}$ and $T^1_1 = \{G\}$, based on their connectivity. The threshold is increased by 1 and the process is repeated. All members of T^1_0 equal to $h_{min} + 1$ are placed in a new node, $C^0_1 = \{A, C, E\}$, and all others placed into temporary nodes based on their connectivity. This process is repeated until the maximum value (h_{max}) is reached. The resulting max-tree is the filtered simply by visiting each node and applying a constraint. In the case of the area attribute, only the leaf nodes need to be checked. If a node fails to meet the criteria, then it is removed and absorbed into its parent. So long as the parent remains a leaf node, then it will continuously be checked and pruned until if meets the criteria. This is an efficient method to code as given in Listing 6.6, which is improved by the use of hierarchical queue implementation [135]. The listing uses three functions to manipulate the queue:

- Queue_add(*h*,*p*),
    - Adds the pixel *p* of value *h* to the queue at a priority of *h*.

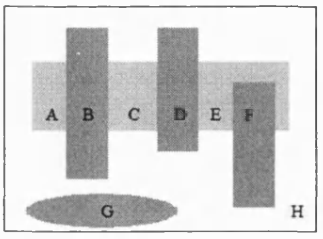
- Queue_first(*h*),
    - Extracts the first pixel of the queue at pixel *h*.

- Queue_empty(*h*),
    - Returns true if the queue at level *h* is empty.

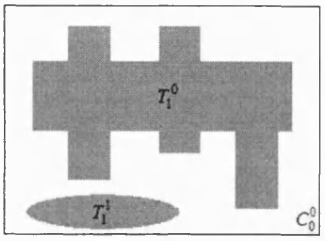
The algorithm also uses some extra notation; number_nodes(*h*) which defines the number of nodes at level h. This is initialised to be 0 at the start of the tree creation. Orig_Image(*p*), which gives the original value of pixel *p*. Status(*p*) stores information about the pixels status which can be one of three values:

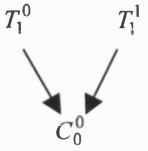
- Not_analysed,
- In_the_queue,
- Equal to *k*, which indicates that it is assigned to node *k* of level *h*.

The process of extracting nodes is applied recursively. This is started by finding the first pixel with the lowest value h_{min}.

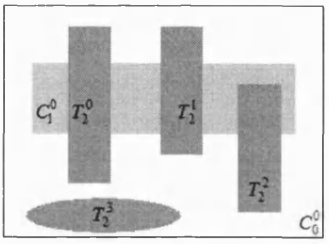


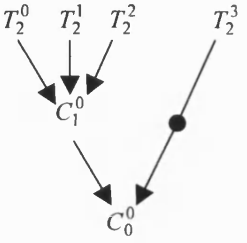
a) Input image



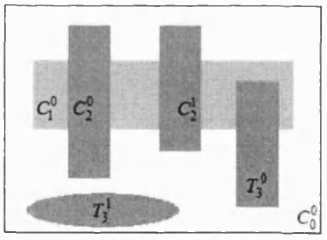


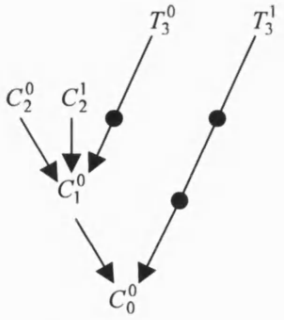
b) Threshold image with *h=0* and the corresponding max-tree.



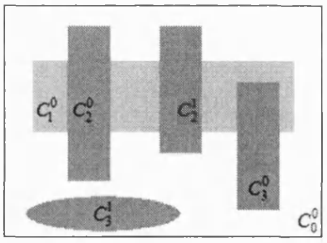


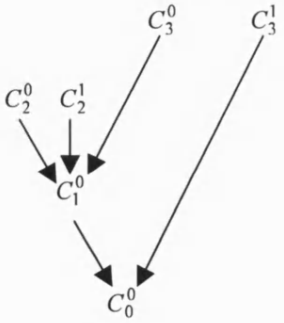
c) Threshold image with *h=1* and the corresponding max-tree.





d) Threshold image with *h=2* and the corresponding max-tree.





e) Threshold image with *h=3* and the corresponding max-tree.

Figure 6.16: A greyscale Max-tree.

```
//The main rountine
void main(void){
        for (h=0 to 255)
                number_nodes(h) = 0        //defines the number of nodes at level h.
        for (p=0 to number_elements_in_image)
                Status(p) = not_analysed
        for (p=0 to number_elements_in_image)
                if (Orig_image(p) = h_min){
                        flood(p)
                        return 0
                }
        }
        return –1
}

//Flooding rountine
unsigned char flood(unsigned char h){
        while (queue_empty(h) = false){
                p = queue_first(h)
                Status(p) = number_nodes(h)
                For all neighbours, n, of p do:
                        If (Status(n) = not_analysed){
                                Queue_add(Orig_image(n),n)        //Add neighbour n to the queue
                                                                  //at level Orig_image(n).
                                Status(n) = in_the_queue
                                Node_at_level(Orig_image(n)) = true
                                Node_at_level(Orig_image(p)) = true
                                If (Orig_image(n) > Orig_image(p)){
                                        m = Orig_image(n)
                                        do{
                                                m = flood(m)
                                        }while (m = h)
                                }
        }
        number_nodes(h) = number_nodes(h) + 1
        m = h – 1

        while (m≥0 AND node_at_level(m) = false){
                m = m – 1
        }
```

Listing 6.6: Max-tree creation algorithm.

```
        if (m≥0){
                i = number_nodes(h) – 1
                j = number_nodes(m)
        }else{
                //node has no father, i.e. it is the root
        }
        node_at_level(h) = false
        return m
}
```

Listing 6.6 continued.

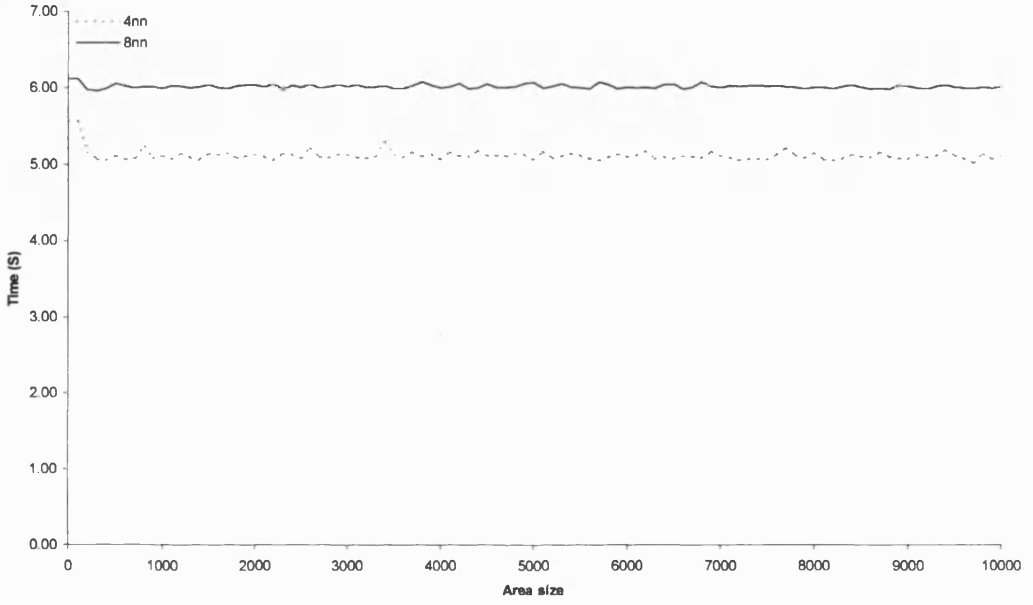


Figure 6.17: Timing results for the Max-tree method.

As can be seen from Figure 6.17, above a certain area size, 700 and 3400 for 8nn and 4nn connectivity respectively, the max-tree method is significantly faster than the pixel queue method. Also, although the image contents do affect the speed of the max-tree to a certain degree, the area size has little impact on the speed, which remains almost constant at approximately 5 seconds and 6 seconds for 4nn and 8nn connectivity respectively. There are several different techniques, which can be used in filtering the max-tree such as Virterbi, Direct and Subtractive [136]. Like the pixel queue method, using 4nn connectivity is faster than 8nn connectivity over the range of images used. At an area size of 3000, the max-tree is on average 1.1 times faster than Vincent's using 4nn and 2.5 times faster using 8nn. The implementation of the max-tree presented by Meijster and Wilkinson gives an increase of approximately 3.2 times faster than their pixel-queue method [132]. Without knowing what connectivity is being used, it is difficult to compare the performance. The dual of a max-tree is a Min-tree, produces the output of the AC filter (see section 4.6.1).

6.1.5 Wilkinson's Method

Tarjan introduced an efficient technique for manipulating disjoint sets called the Union-Find method [137]. This method has been adapted for the AO (see section 4.6.1) operator by Meijster and Wilkinson and later adapted to attribute openings by Wilkinson and Roderdink [132], [138]. The two previous methods both work by flooding pixels and also only process a single extrema at a time. This method has the advantage processes multiple extrema at the same time. The pixels of the image are sorted using a radix sort. This puts the pixels in order according to their grey level. Each pixel is then processed, starting with the highest value. Rooted trees are used in a similar way to the Max-tree method in order to keep track of which pixels belong to which region. The tree is kept in an array, parent, which has the same number of elements as the image. The array is initialised to a value of $-(\lambda+1)$ where λ is the target area size. A value lower than zero and greater than the initialised value indicated an active root. A value equal to or greater than zero indicates a node, which belongs to a region. The value of parent at this location points to the parent location of this pixel. Five basic routines are used to implement this method; the main routine is shown in Listing 6.7:

- RadixSort (see Listing 6.8)
  - This sorts the pixels of the image in order of their greyscale value. The highest values are placed at the right end of the array and the lowest values at the left end.

- MakeSet(x) (see Listing 6.9)
  - Create a new singleton set, $\{x\}$, which assumes that x is not a member of any other set.

- FindRoot(x) (see Listing 6.10)
  - This function finds the root of the tree containing the pixel x.

- Union(x,y) (see Listing 6.11)
  - Forms the union between the two sets that contain pixels x and y.

- Criterion(x,y) (see Listing 6.12)
  - This function checks to see if pixels x and y belong to the same tree and if they do not, it also checks the attribute of the two pixels.

The timing results are shown in Figure 6.18, which shows a significant improvement upon the previous methods. Timing measurements were made on the test system, an AMD XP1800 running Windows XP with 512MB of RAM (see section 6), with measurements being taken at intervals of 100 in area size starting at and area size of 1 and ending at 10001. Again the 4nn method is slightly faster than the 8nn. In addition this method is slightly dependant upon the image content and attribute size used. At an area size of 8000, this method is on average 25.6 times faster than the pixel queue using 4nn and 46.3 times faster using 8nn. The implementation presented by Meijster and Wilkinson gives

an increase of approximately 3.8 times faster than their pixel-queue method [132]. Without knowing what connectivity is being used, it is difficult to compare the performance. However the methods implemented for this thesis do give a much greater increase in performance when compared with the pixel-queue method than those presented by Meijster and Wilkinson. As can be seen, this method is a significant improvement over the two previous methods. Like the max-tree, the time taken to process to a given size is almost a constant value of 0.23s and 0.33s for 4nn and 8nn connectivity respectively. However it can also be seen that the 8nn connectivity fluctuates more than the 4nn connectivity, although only by about 20mS.

6.2 Attribute Morphology

Whilst the area attribute is of use, it may not necessarily be the best for every application. For example, in segmentation, the complexity or shape of a region may be a better attribute or for video the motion. Breen and Jones modified the pixel queue method so that any attribute may be used [79]. In addition both the Max-tree method and Wilkinson's method can also process other attributes [138].

```
void main(void){
        RadixSort();
        for (int s=Length(sorted); s >= 0; s--){
                pix = sorted[s];
                MakeSet( pix );
                for all neighbours, n, of pix, do:
                        if (( Image[pix] < Image[n]) || ((Image[pix] == Image[n]) && (n<pix)))
                                Union(n,pix)
        }

        //Resolving scan in reverse sort order

        for (s=0;s<Length(sorted);s++){
                pix = sorted[s];
                if (parent[pix] >= 0)
                        parent[pix] = parent[parent[pix]];
                else
                        parent[pix] = Image[pix];
        }
}
```

Listing 6.7: The main body of Wilkinson's method.

```
void RadixSort( void )
{
        //Build a histogram of the image
        For i=0 to (Number Elements – 1)
                Histogram[ Image[ i ] ] = Histogram[ Image[ i ] ] + 1

        //Make a running total
        For i=1 to (Number levels –1)     //256 for an 8 bit image.
                Histogram[ i ] = Histogram[ i ] + Histogram[ i-1 ]

        //Create the sorted list.  The highest value will be on the right
        For i=0 to (Number Elements – 1)  {
                Sorted[ Histogram [ Image [ i ] ] - 1 ] = i
                Histogram[ Image [ i ] ] = Histogram[ Image [ i ] ] – 1
        }
}
```

Listing 6.8: The radix sort.

```
void MakeSet ( int x){
        parent[x] = -1
}
```

Listing 6.9: The Makeset function.

```
int FindRoot(int x){
        if (parent[x] >= 0)
        {
                parent[x] = FindRoot( parent[x] );
                return parent[x];
        }else{
                return x;
        }
}
```

Listing 6.10: The FindRoot function.

```
bool Criterion(int x, int y){
        return ( ( Image[x] == Image[y]) ||  //The two pixels are the same value, in which case the
                                             //union can then be done
                (-parent[x] < λ));           //or if the area size is less than the target.
}
```

Listing 6.11: The Criteria function.

```
void Union(int n, int p){
        int r = FindRoot(n);
        if (r!=p){
                if (Criteria(r,p)){
                        parent[p] = parent[p]+parent[r];
                        parent[r] = pl
                }else{
                        parent[p] = - λ;
                }
        }
}
```

Listing 6.12: The Union operation.

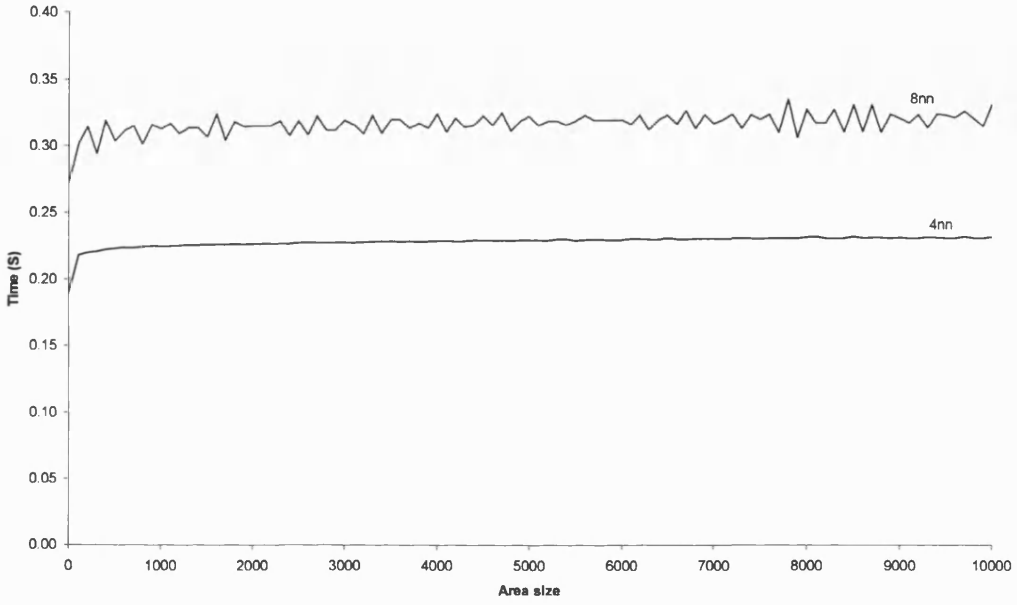


Figure 6.18: Timing results for Wilkinson's method using 50 8bit greyscale, 720 x 576 images. Measured at intervals of an area size of 100 starting at 1 and ending at 10001.

6.2.1 Pixel Queue Algorithm for Attributes

Breen and Jones found that it was relatively simple to convert Vincent's method to process other attributes [79]. Vincent assumed that the attribute, area, is increasing. That is if a set C satisfies the criteria T, then all supersets of C will also satisfy the criteria. Breen and Jones do not make this assumption, which allows for non-increasing criteria to be used. Since the four attributes being used here (see section 6.2.2) are all increasing, the attribute method can be simplified further as given in Listing 6.13. The reader is referred to [79] for the non-increasing version.

6.2.2 Attributes

Four attributes are used for the basic 2D noise evaluation. They are the area, contrast, volume and power. Figure 6.19 shows a comparison of the different attributes and the following sections give a brief description of each of these attributes.

1. Extract the regional maxima (m) of I,
2. For each m of I, do the following:
    a. If the attribute of this maxima is larger that the target size (λ), then move onto the next maxima,
    b. Find the highest greyscale valued neighbour (n) of m,
    c. If the value of n is greater than the value of the maxima, then m is no longer a maxima so move onto the next maxima (go to step 2),
    d. Add pixel n to the maxima m,
    e. Set the value of all pixels belonging to m to be the value of n,
    f. Repeat this procedure, from step a until the attribute of region m is greater than λ.
3. Repeat step 2 until all m have been processed.

Listing 6.13: Attribute Morphology algorithm.

6.2.2.1 Area

The area simply measures the number of pixels within a region. This has been used extensively in image processing. Whilst it is good for some tasks, such as identifying coins from their size, its usefulness for noise reduction and lossless image simplification have yet to be proven.

6.2.2.2 Contrast

The contrast attribute has been used in several applications. It measures the change in intensity of a region which is given by:

$$Contrast(X) = |x_{\max} - x_{\min}| \tag{6.21}$$

where X is the region being grown, $x_{\max}$ is the maximum greyscale value within the region and $x_{\min}$ is the minimum greyscale value within the region. This attribute is motivated partially by the fact that the HVS (see chapter 3) cannot detect small changes in contrast. However, noise can contain high peaks, which would require a large attribute value, which, in turn would remove much of the detail of the image. Thus this attribute may work well with low noise but not high noise, which is to be proven. Contrast can be implemented by using the reconstruction operators, which can be implemented very efficiently (see section 6.1.1).

6.2.2.3 Volume

The two previous attributes both contain desirable qualities. The HVS (see chapter 3) cannot detect large changes in intensity over a small area or small changes over a large area. The volume combines both the area and contrast attributes together by measuring the volume removed by a region. The

volume of a region X is given by equation 6.22, which can be calculated on the fly as shown in Figure 6.19.

$$Volume(X) = \sum_{\forall x \in X} |x_i - v| \tag{6.22}$$

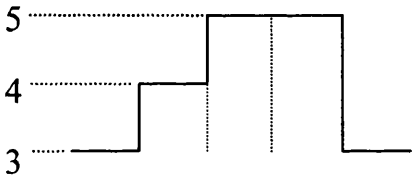
where v is the new value of the region.

6.2.2.4 Power

The three attributes discussed so far have all been used prior to this research. For preprocessing an image for psychovisual redundancy, an attribute is required that is best matched to the HVS (see chapter 3). It is proposed that measuring the power that a region removes from the original image will be proved to accomplish this. The power is given by:

$$Power(X) = \sum_{\forall x \in X} |x_i - v|^2 \tag{6.23}$$

where v is the new value of the region. This attribute is motivated by the fact that the $(\text{change in intensity})^2 \times \text{area}$ corresponds to the response of the HVS to incident radiation. Thus this attribute should be a closer match to the HVS than any other.



a) Original signal

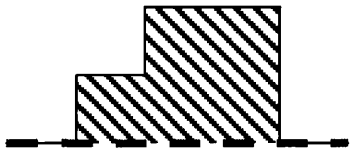
Area = 2

Contrast = 5 – 4 = 1

Volume removed = $2 \times (5-4) = 2$

Power removed (shaded) is $2 \times (5-4)^2 = 2$.

b) Grow the region.



Area = 3

Contrast = 5 – 3 = 2

Volume removed = $2+3 \times (4-3) = 5$

Power removed (shaded) is $2 \times (5-3)^2 + (4-3)^2 = 9$.

c) Grow the region.

Figure 6.19: Comparison of different attributes.

6.3 Filter Structures

The two filtering structures that are most commonly used, the AF and the ASF were defined in section 4.6.2. However, Vincent states that the difference between AF and ASF filters is negligible whilst others report the opposite [57], [139]. Chapter 7 investigates the differences between these filters both objectively and subjectively using noise reduction and psychovisual redundancy respectively to show conclusively the differences between the two filtering structures.

6.3.1 Efficient ASF Implementation

Due to the fact that the ASF is applied iteratively, it is relatively slow when compared to the AF (see section 4.6.2). For example, to perform an AOC (see section 4.6.1) with an area size of 100, just two operations are required with the AF filter structure, an open and a close to an area size of 100. However, the ASF filter structure (see section 4.6.2) would require 198 operations (99 openings and 99 closings). This is shown more clearly by the timing measurements taken over 50 8bit greyscale, 720 x 576 pixel images, using the test system comprising of an AMD XP1800 processor with 512MB RAM running Windows XP (See section 6). The results of the timing experiments are shown in Figures 6.20 and 6.21 for 4nn and 8nn connectivity respectively. This shows the previous methods, the pixel queue (see section 6.1), the Max-tree (see section 6.1.4) and Wilkinson's method (see section 6.1.5) when used as an ASF.

A reduced complexity filter for an ASF filter using symmetric dynamics was proposed by Vachier and Vincent [140], [141]. Research carried out for this thesis has independently developed a similar algorithm but generalised so that any attribute may be used. The algorithm (see Listing 6.14) is a variation of the pixel queue algorithm, which works by first extracting all of the regional minima and maxima. By definition, regional maxima cannot also be regional minima, so both may be extracted simultaneously. To make this more efficient, rather than re-obtaining the extrema every time the attribute is increased, two lists of regions is stored, one for the maxima regions and one for the minima. Each stored region contains information such as the attribute value and pixel locations that are part of the region. To keep the property of the ASF, first all maximas of size 1 are grown and then minima's of size 1 are then grown, the attribute is then increased and the operation repeated until the target attribute limit is reached. By using the two lists, the maxima list can be transcended searching for all regions with a given attribute and then the minima region list is processed. When a region is no longer an extrema, it is removed from the list. Thus this avoids the need to reprocess the entire image every time as other methods do and thus increases the efficiency of the ASF filter structure. In addition, to ensure that the new algorithm works correctly, a morphological Matlab toolbox called '**SDC Morphology Toolbox**' and source code supplied by *Michael Wilkinson* have been used to confirm the filter output [129]. Several images were filtered to various area sizes using the supplied code. The resultant images were than compared to the output of the new efficient method, which confirmed that the results were identical, thus showing that the new method worked correctly.

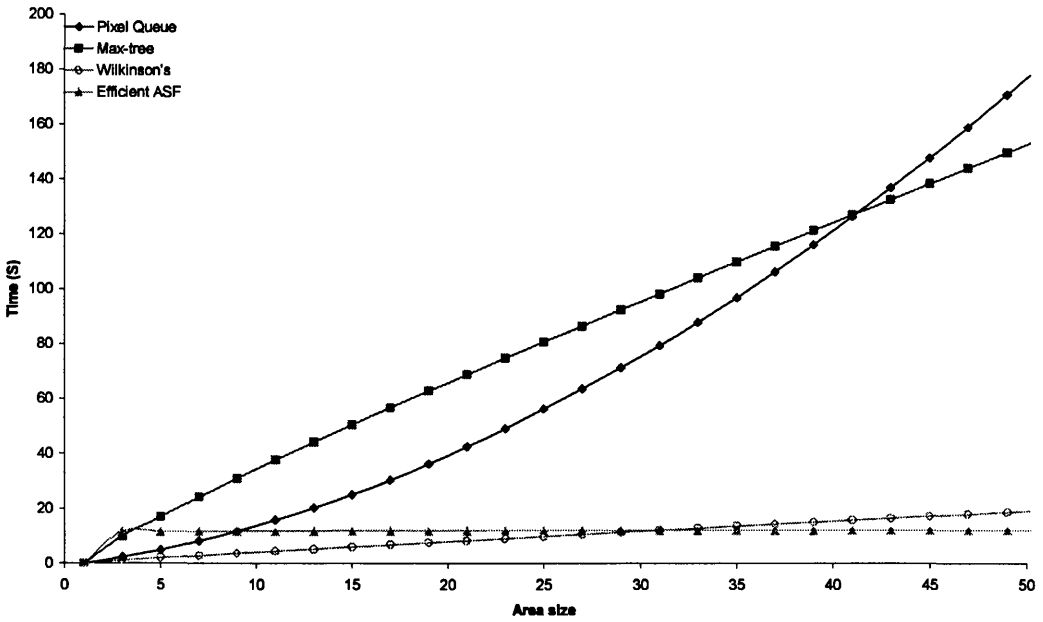


Figure 6.20: Timings (in Seconds) for ASF implementations using 4nn connectivity, 50 8bit greyscale images with a size of 720 x 576. Measurements have been made at area size intervals of 2, starting at 1 and ending at 51.

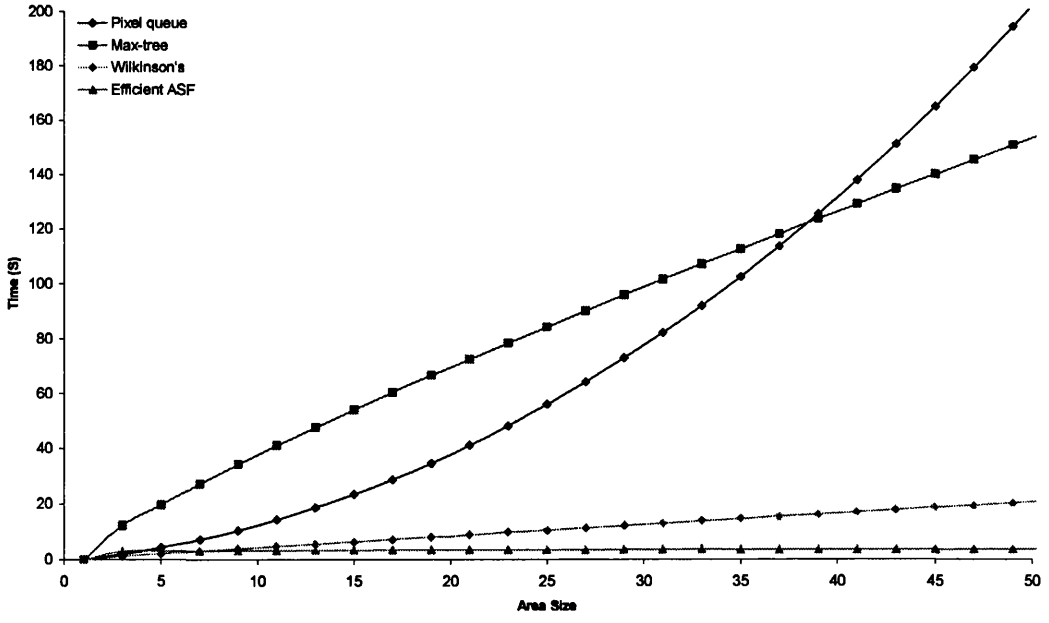


Figure 6.21: Timings (in Seconds) for ASF implementations using 8nn connectivity, 50 8bit greyscale images with a size of 720 x 576. Measurements have been made at area size intervals of 2, starting at 1 and ending at 51..

1. Extract the regional maxima (m_{max}) of I,
2. Extract the regional minima (m_{min}) of I,
3. Starting from $\lambda = 1$, do the following
    c) For each m_{max} of I with an area of λ, do the following:
        - Find the highest greyscale valued neighbour (n) of m_{max},
        - If the value of n is greater than the value of m_{max}, then m_{max} is no longer a maxima so deactivate the region and move to the next maxima (go to step a),
        - If n is a member of a minima region then
            - Let the region containing pixel n swallow the current region, m_{max}.
            - Set the value of all pixels belonging to the merged region to be the value of n.
        - Otherwise
            - Add pixel n to the current m_{max},
            - Set the value of all pixels belonging to the current m_{max} to be the value of n.
        - Repeat this procedure, from step a until all maximas with an area of λ have been processed.
    d) For each m_{min} of I with an area of λ, do the following:
        - Find the lowest greyscale valued neighbour (n) of m_{min},
        - If the value of n is greater than the value of m_{min}, then m_{min} is no longer a minima so deactivate the region and move to the next minima (go to step b),
        - If n is a member of a minima region then
            - Let the region containing pixel n swallow the current region, m_{min}.
            - Set the value of all pixels belonging to the merged region to be the value of n.
        - Otherwise
            - Add pixel n to the current m_{min},
            - Set the value of all pixels belonging to the current m_{min} to be the value of n.
        - Repeat this procedure, from step a until all minimas with an area of λ have been processed
    e) If $\lambda \neq \lambda_{target}$, increase λ by 1 and repeat the operation from a.

Listing 6.14: The efficient ASF algorithm.

The results, shown in Figures 6.20 and 6.21 for 4nn and 8nn connectivity respectively, shows that the efficient ASF implementation has a significant increase in speed compared to the standard pixel queue and Max-tree methods. The 4nn shows an average increase of 15.63 times faster than the pixel-queue method and the 8nn gives a 45.76 times faster. The results show the max-tree method performing worse than the pixel-queue algorithm until an area size of 30 for the 4nn using the Barbara image. After this point, the max-tree method performs better than the pixel-queue. The time taken for the

efficient ASF is relatively constant. For the Barbara image, the efficient ASF takes an average of 8.35 seconds for 4nn and 3.29 seconds for 8nn regardless of the area size used. To show this difference further, 50 test images were used to measure the performance of the ASF implementations as shown in Figures 6.20 and 6.21 for 4nn and 8nn connectivity respectively. The average time is measured at area size intervals of 2 starting from a size of 1. Both 4nn and 8nn connectivity show that below an area size of 6, Wilkinson's method performs best, but at larger area sizes show that the new efficient ASF implementation performs significantly faster than other methods. To illustrate this increase further, Table 6.3 shows the average execution time for an area size of 47, which is relatively small considering there are 414720 pixels in the images. This shows that for the 3 standard methods, 4nn connectivity is faster than 8nn connectivity, where as the new efficient ASF shows that the 8nn connectivity is faster. It should be noted however, that although the algorithm is more efficient than other methods, the implementation has not been fully optimised and may account for the significant difference in speed between the 4nn and 8nn connectivity. Despite this, the new efficient method is still faster then all other methods, and is 13.1 and 48.2 times faster than the pixel queue method for 4nn and 8nn connectivity respectively. Thus this has shown that the new efficient ASF algorithm is significantly faster than the current methods, and may be improved by a more optimised implementation.

Connectivity	Execution time (seconds)			
	Pixel queue	Max-tree	Wilkinson's	Efficient ASF
4nn	159.10	144.21	18.05	12.13
8nn	179.89	145.99	19.57	3.73

Table 6.3: Average execution time (in seconds) for an area size of 47 taken over 50 8bit greyscale images with a size of 720 x 576 pixels.

6.4 Conclusion

This chapter has shown several different ways in which AM can be efficiently implemented. Timing measurements have shown how efficient the different methods are. The implementation has been extended to attribute morphology and the two filter structures, AF and ASF have been discussed. The ASF has been looked at in detail and a new efficient implementation has been devised, which significantly increases the execution of the ASF filter. This is supported with real world timing measurements, which show the new efficient ASF method has an almost constant execution time and is significantly faster than other methods in use.

A morphological Matlab toolbox called **'SDC Morphology Toolbox'** and source code supplied by *Michael Wilkinson* have been used to ensure that the implementation of filters written for this thesis produce the correct outputs [129]. That is that they all produce identical results for the same filters and parameters. Whilst checking this, the SDC Morphology toolbox was actually found to contain errors, which were confirmed with the programmers, who have since corrected the errors.

7 Preprocessor Development

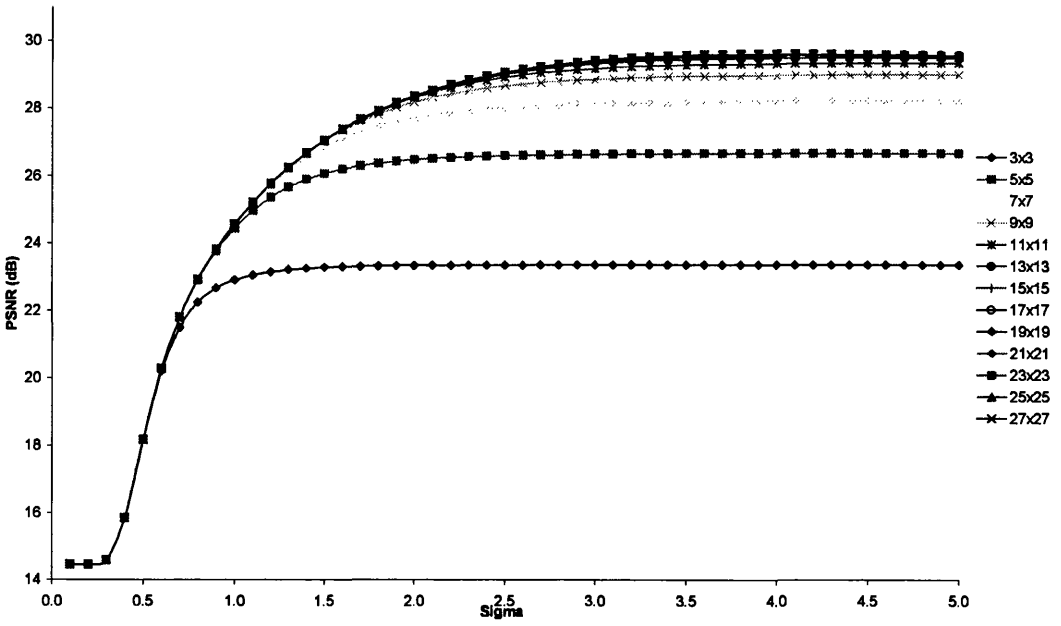
This chapter details the implementation of a preprocessing system for images in several stages. Section 7.1 develops a noise reduction system, which is evolved further in chapter 9 for application to video sequences. The second section, section 7.2, implements a psychovisually lossless system, which starts by using standard area-morphology to filter images. These are then compressed and decompressed to obtain a rate-distortion graph, which will then show how good area morphology is for image simplification before proceeding with further development. The third stage, described in section 7.3, shows the investigation of an intermediate morphology processing system, whereby the entire image is filtered and not just the minima and maxima regions. To evaluate the compressibility of the filtered images, two CODEC's are used for the image evaluation, JPEG and JPEG 2000. JPEG is a close approximation to an intra frame in MPEG and JPEG 2000 is built using wavelets so will give a good approximation to the Bath Wavelet Video (BWV) CODEC [142]. Also, these two CODEC's use the two most widely adopted methods. That is the DCT (JPEG) and wavelet (JPEG 2000) transforms.

7.1 Image Noise Reduction

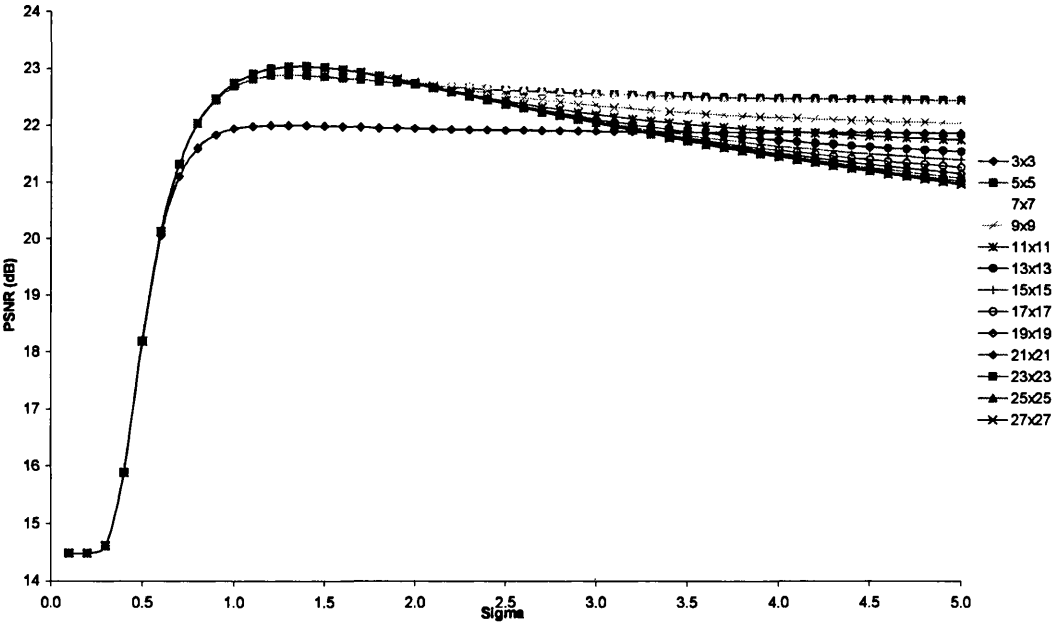
One of the key features of any preprocessing system is the ability to remove or reduce noise. AWGN (see section 5.1.1) is used as it is considered to be a worst-case condition. Five test images; Barbara, Barbara 2, Boats, Goldhill and Simulated are corrupted with noise of varying amounts, which give a PSNR of 14dB, 21dB, 28dB and 35dB when compared to the original images. The resultant noisy images are then filtered to reduce the noise present and thus make it easier to compress. This noise reduction is attempted in two parts. The first part uses standard Gaussian filtering (see section 5.3.1.2), as this is a well known and understood filtering method. The second part evaluates morphological filters using the AF and ASF filter structures (see section 4.6.2), the area, contrast, volume and power attributes (see section 6.2.2) and 4nn and 8nn connectivity. In addition, the morphological filtering is compared to the Gaussian filtering method (see section 5.3.1.2).

7.1.1 Gaussian Filtering

The Gaussian filtering method (see section 5.3.1.2) is used as a reference point from which the morphological filtering methods can be compared to. The Gaussian filter has two variables that affect the weights of the neighbours, σ, and the size of the mask. The amount of noise and image content both affect what values these should be set to. The only way to find the optimum, the set of variables that produces the best PSNR, is to iterate through a range of variable values. Thus the mask size is varied from a 3 x 3 mask up to a 27 x 27 mask and the value of σ is varied from a value of 0.1 to 5 in steps of 0.1. Results of this operation for the Barbara and Simulated images are shown in Figures 7.1 – 7.4 for 14dB, 21dB, 28dB and 35dB of noise respectively. Results for the Barbara 2, Boats and Goldhill images are given in Appendix B, Figures 13.1 – 13.12.

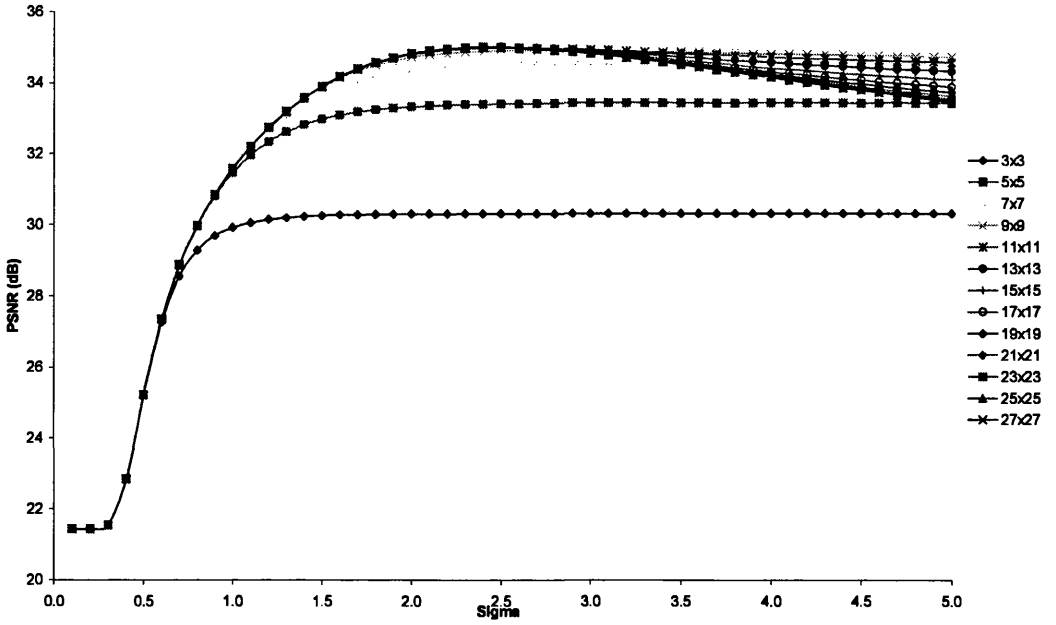


a) Simulated image (128 x 128 pixels, 8bit greyscale).

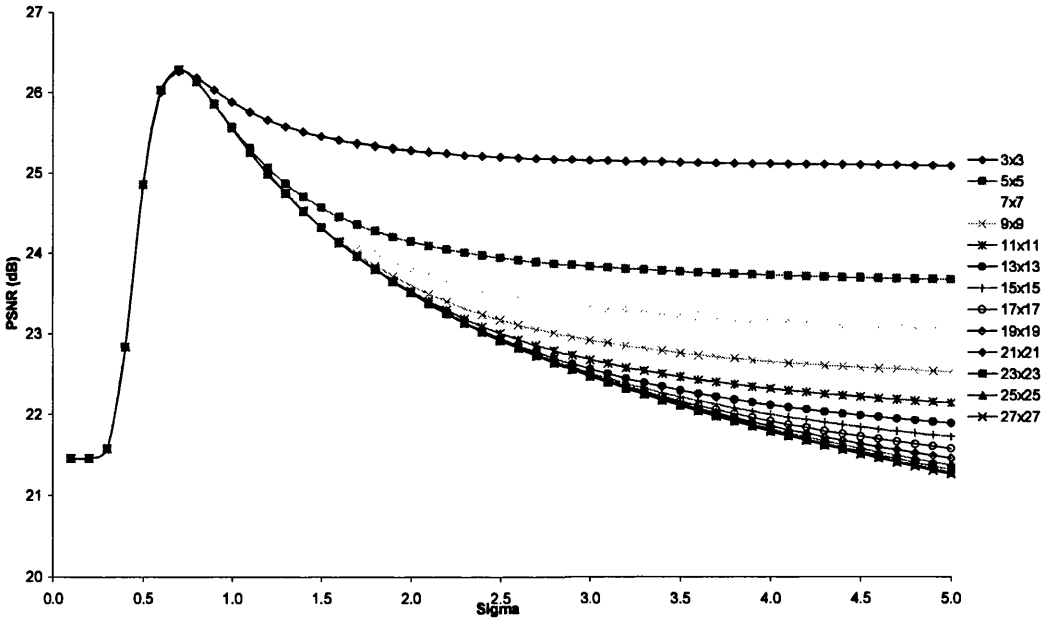


b) Barbara image (720 x 576 pixels, 8bit greyscale).

Figure 7.1: Gaussian results showing PSNR against sigma and mask size with 14dB of noise.

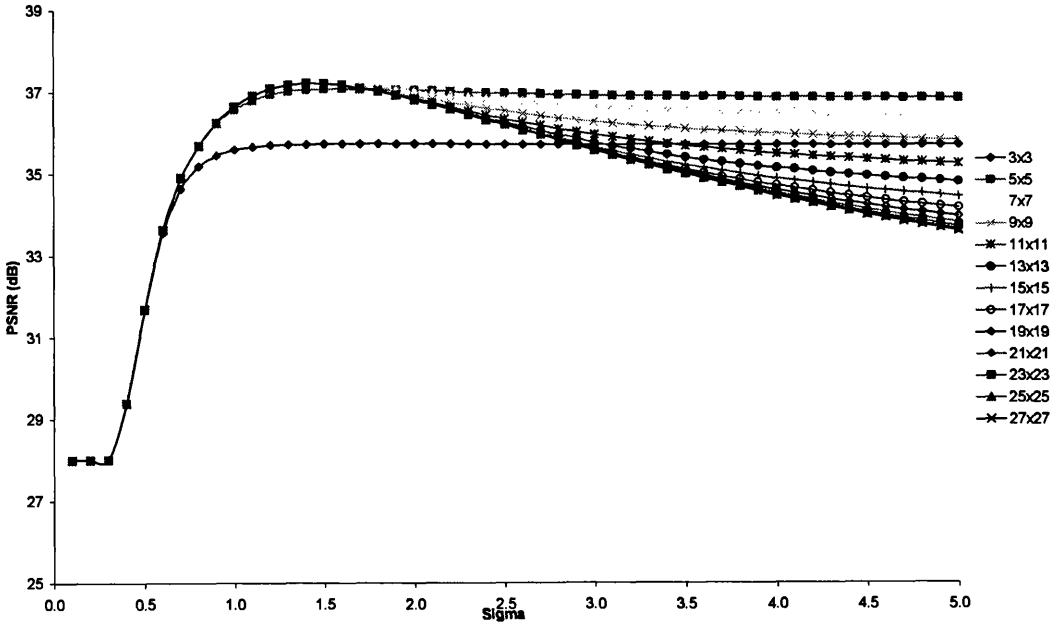


a) Simulated image (128 x 128 pixels, 8bit greyscale).

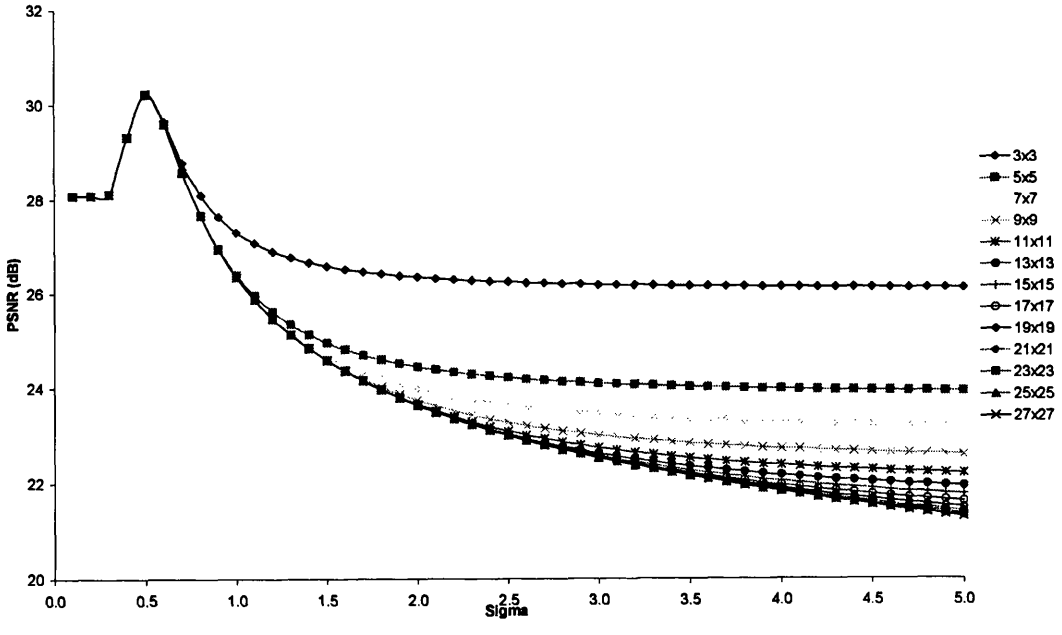


b) Barbara image (720 x 576 pixels, 8bit greyscale).

Figure 7.2: Gaussian results showing PSNR against sigma and mask size with 21dB of noise.

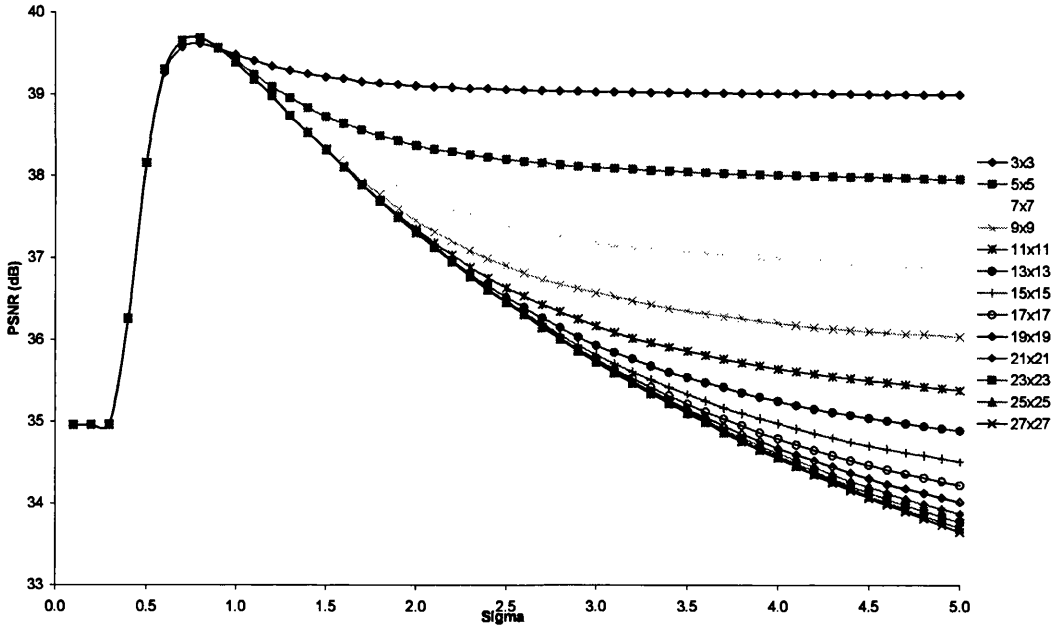


a) Simulated image (128 x 128 pixels, 8bit greyscale).

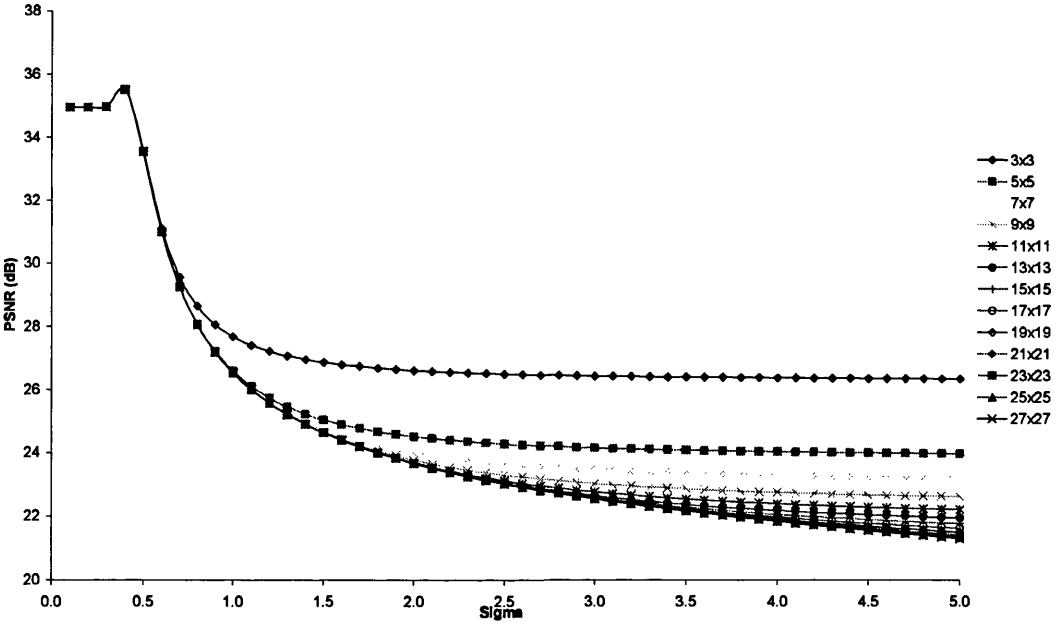


b) Barbara image (720 x 576 pixels, 8bit greyscale).

Figure 7.3: Gaussian results showing PSNR against sigma and mask size with 28dB of noise.



a) Simulated image (128 x 128 pixels, 8bit greyscale).



b) Barbara image (720 x 576 pixels, 8bit greyscale).

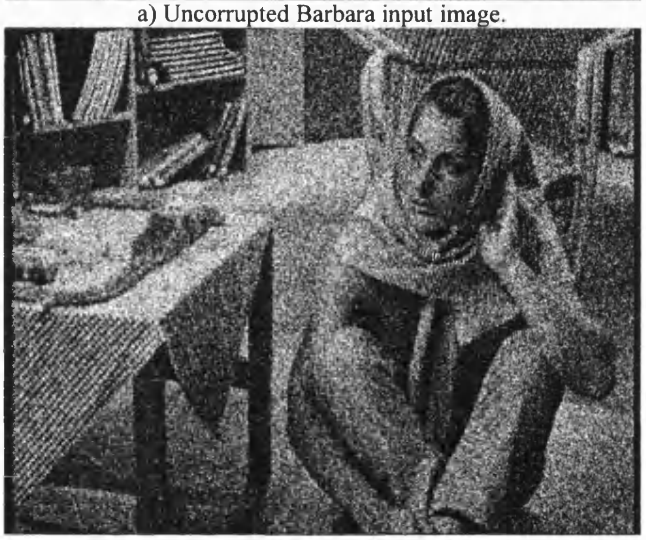
Figure 7.4: Gaussian results showing PSNR against sigma and mask size with 35dB of noise.

As can be seen from the graphs, the PSNR performance increases with σ until a certain value is reached, after which the PSNR performance decreases. In addition, as the amount of noise increases, the point where σ peaks, moves up showing that the higher the amount of noise present, the wider the Gaussian function needs to be. This is so that more of the neighbours surrounding the pixel can have a higher influence on the output and is shown more clearly by highlighting the optimum filter values as shown in Table 7.1 (the results for the Barbara 2, Boats and Goldhill images are shown in Appendix B, Table 13.1). Coinciding with this is the fact that the optimum mask size increases with noise, again allowing more of the neighbours to influence the output of the filter. Table 7.1 shows that the Gaussian filter, although relatively simple, effectively removes a significant amount of noise. For example, 8.56dB of noise can be removed from the Barbara image with 14dB of noise (see Figure 7.5). However, this filter is not as effective with a low amount of noise as only 0.55dB of noise is removed from the Barbara image with 34.96dB of noise present (see Figure 7.6). The Barbara 2, Boats and Goldhill images all show the same trend (see Table 13.1). Figures 7.5 and 7.6 show the input and output of the optimum filters for 14dB and 35dB of noise respectively. Although a significant amount of noise is added, much of the detail can still be seen, for example the lines on the scarf, and the pattern on the table cloth. However, after filtering much of the original detailed is removed.

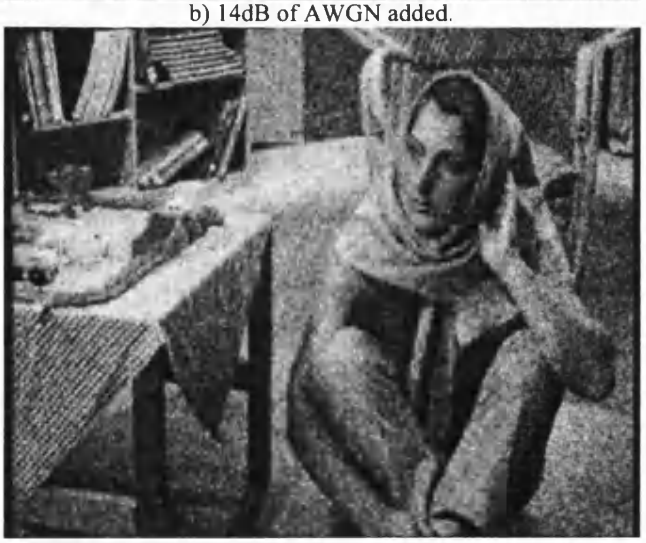
Image and noise level (dB)	Sigma	Mask Size	PSNR (dB)
Barbara 14.48dB	1.4	15 x 15	23.04
Simulated 14.47dB	4.0	27 x 27	29.62
Barbara 21.46dB	0.7	7 x 7	26.29
Simulated 21.44dB	2.4	21 x 21	35.03
Barbara 28.09dB	0.5	3 x 3	30.24
Simulated 28.01B	1.5	7 x 7	37.24
Barbara 34.96dB	0.4	3 x 3	35.52
Simulated 34.96dB	0.8	9 x 9	39.69

Table 7.1: The optimum sigma values and mask sizes for noise removal using the Gaussian filter and their performance gain in PSNR.

a) Uncorrupted Barbara input image.

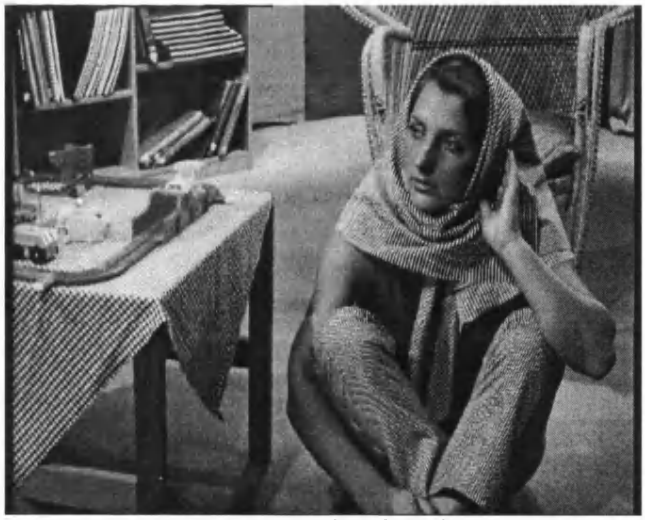


b) 14dB of AWGN added.



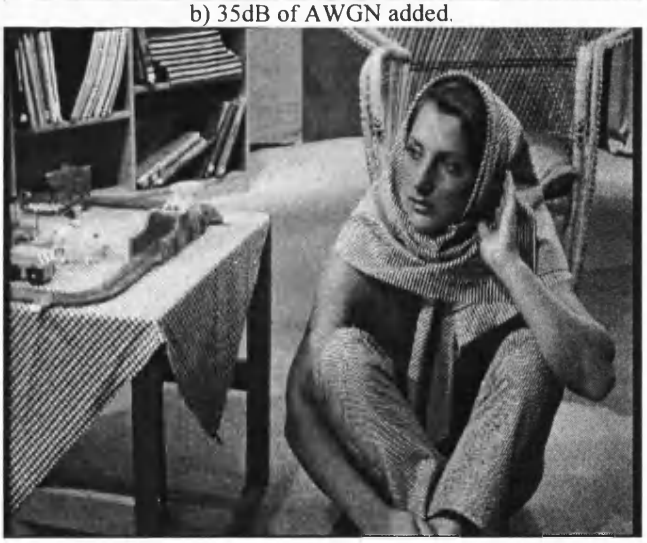
c) Optimum Gaussian filter using a $\sigma = 1.4$ and a mask size of 15 x 15, which gives a PSNR of 23.04dB (an increase of 8.56dB).

Figure 7.5: Filtering the Barbara image (8bit greyscale at 720 x 576 pixels) using the optimum Gaussian filter.



a) Uncorrupted Barbara input image.

b) 35dB of AWGN added.



c) Optimum Gaussian filter using a $\sigma = 0.4$ and a mask size of 3 x 3, which results in a PSNR of 35.52dB (an increase of 0.56dB).

Figure 7.6: Filtering the Barbara image (8bit greyscale at 720 x 576 pixels) using the optimum Gaussian filter.

7.1.1.1 Compression Results

Although the Gaussian filtering method successfully removes noise from the images, the resultant filters images may not necessarily compress any better. Hence two still image compression CODEC's, JPEG and JPEG 2000, are used to evaluate how well the optimum filtered images compress. The noisy images are filtered using the optimum solutions found, which are then compressed to a ratio of 20:1 (or 0.4 bpp). A comparison, using the PSNR is then made between the output of the CODEC and the original uncorrupted input image. The results for the Barbara and Simulated images are shown in Table 7.2 (the results for the Barbara 2, Boats and Goldhill are in Appendix B, Table 13.2). These show that when using the JPEG CODEC, the original noise free Barbara image has a PSNR of 27.32dB, however, when the noisy version (14dB) is compressed, the resultant output has a PSNR of only 18.59dB. This shows both that noise significantly degrades the compressibility of an image, but also that the CODEC can reduce noise itself, in this case the CODEC has removed 4.11dB of noise.

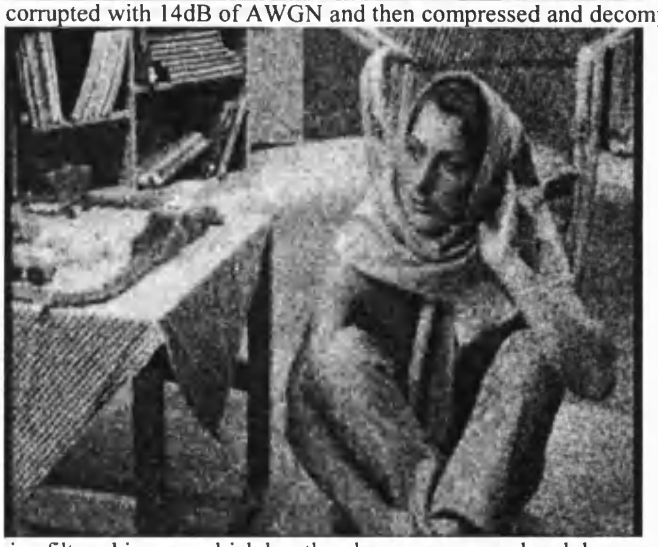
The filtered versions of the Barbara image shows that the output of the CODEC is only improved when there is a lot of noise present. The Barbara 2, Boats and Goldhill images all show the same trend, thus Gaussian filtering will remove noise from an image relatively well, but it will not improve the compressibility of images with a low amount of noise (21dB or above for the majority of images). Figure 7.7 and 7.8 show the output of the JPEG and JPEG 2000 CODEC's for the Barbara image with no noise, 14dB of noise added and the optimum filtered noisy image. This shows that the unfiltered image (see Figure 7.7b) is very blocky, which is due to the noise. However the filtered version (see Figure 7.7c) show much less blockiness. The output of the JPEG 2000 CODEC (see Figure 7.8) shows similar results, but with ringing and drop outs being more dominant in the unfiltered noisy image. Figures 7.9 and 7.10 show the same but using 35dB of noise instead. However, the difference between filtered and unfiltered much less visible.

Image and noise level (dB)	PSNR (dB)			
	Un-filtered images		Optimal filtered images	
	JPEG	JPEG 2000	JPEG	JPEG 2000
Barbara (Original noise free)	27.32	31.16		
Simulated (Original noise free)	45.02	55.33		
Barbara 14.48dB	18.59	17.67	22.11	22.29
Simulated 14.47dB	21.31	19.93	15.58	15.63
Barbara 21.46dB	23.73	24.49	23.82	24.29
Simulated 21.44dB	27.48	25.78	15.93	15.98
Barbara 28.09dB	26.45	28.87	26.16	27.84
Simulated 28.01B	33.71	33.14	14.52	14.57
Barbara 34.96dB	27.20	30.49	25.65	27.15
Simulated 34.96dB	39.14	39.57	14.02	14.10

Table 7.2: PSNR values for the output of the JPEG and JPEG 2000 CODEC's using the optimum Gaussian filtered images as the input. The CODEC outputs are compared to the original noise free image.

a) Uncorrupted image compressed and decompressed.

b) Image corrupted with 14dB of AWGN and then compressed and decompressed.

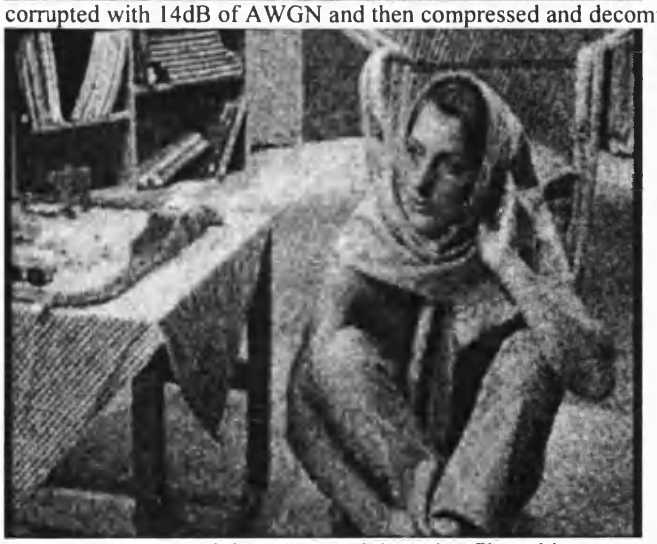


c) Gaussian filtered image, which has then been compressed and decompressed.

Figure 7.7: Barbara image (720 x 576, 8bit greyscale) compressed using the JPEG CODEC with a compression ratio of 20:1 (0.4bpp) and 14dB noise.

a) Uncorrupted image compressed and decompressed.

b) Image corrupted with 14dB of AWGN and then compressed and decompressed.

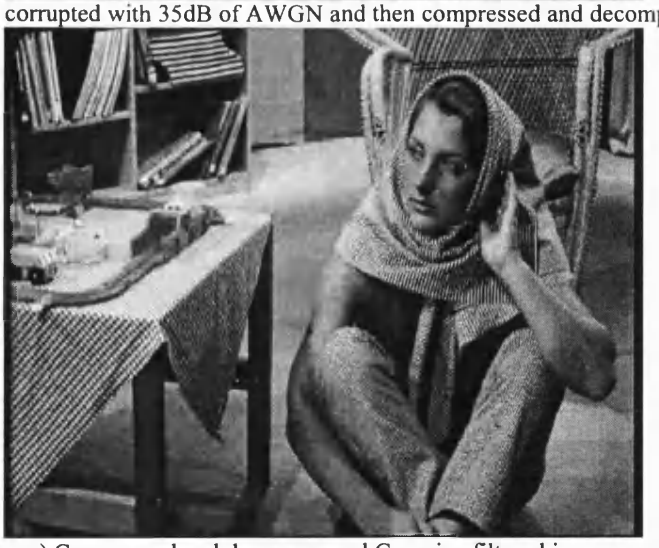


c) Compressed and decompressed Gaussian filtered image.

Figure 7.8: Barbara image (720 x 576, 8bit greyscale) compressed using the JPEG 2000 CODEC with a compression ratio of 20:1 (0.4bpp) and 14dB of noise.

a) Uncorrupted Barbara image compressed and decompressed.

b) Image corrupted with 35dB of AWGN and then compressed and decompressed.



c) Compressed and decompressed Gaussian filtered image.

Figure 7.9: Barbara image (720 x 576, 8bit greyscale) filtered and compressed using the JPEG CODEC with a compression ratio of 20:1 (0.4bpp) and 35dB of noise.

a) Uncorrupted compressed and decompressed image.

b) Image corrupted with 35dB of AWGN and then compressed and decompressed.

c) Compressed and decompressed Gaussian filtered image.

Figure 7.10: Barbara image (720 x 576, 8bit greyscale) filtered and compressed using the JPEG 2000 CODEC with a compression ratio of 20:1 (0.4bpp) and 35dB of noise.

7.1.2 Morphological Filtering

Morphological attribute filters are applied using 4nn and 8nn connectivity; AF and ASF filter structures (see section 4.6.2) and four attributes, area, contrast, volume and power (see section 6.2.2). The optimum attribute values for the morphological filters are found by iterating over a range of attribute values, which is reduced centering around the current optimum value. This is repeated until the range is reduced to one value, which is the optimum value. A comparison of the filtered images are then made by comparing them to the original uncorrupted images. This then indicates the amount of noise that has been removed from the corrupted image. Results for the Simulated and Barbara images are shown in Figures 7.11 – 7.14 for 14dB of noise, Figures 7.15 – 7.18 for 21dB of noise, Figures 7.19 – 7.22 for 28dB of noise and Figures 7.23 – 7.26 for 35dB of noise. Results for Barbara 2, Boats and Goldhill images are given in Appendix B, Figures 13.13 – 13.36. The graphs show the relationship between the attribute value and the amount of noise removed.

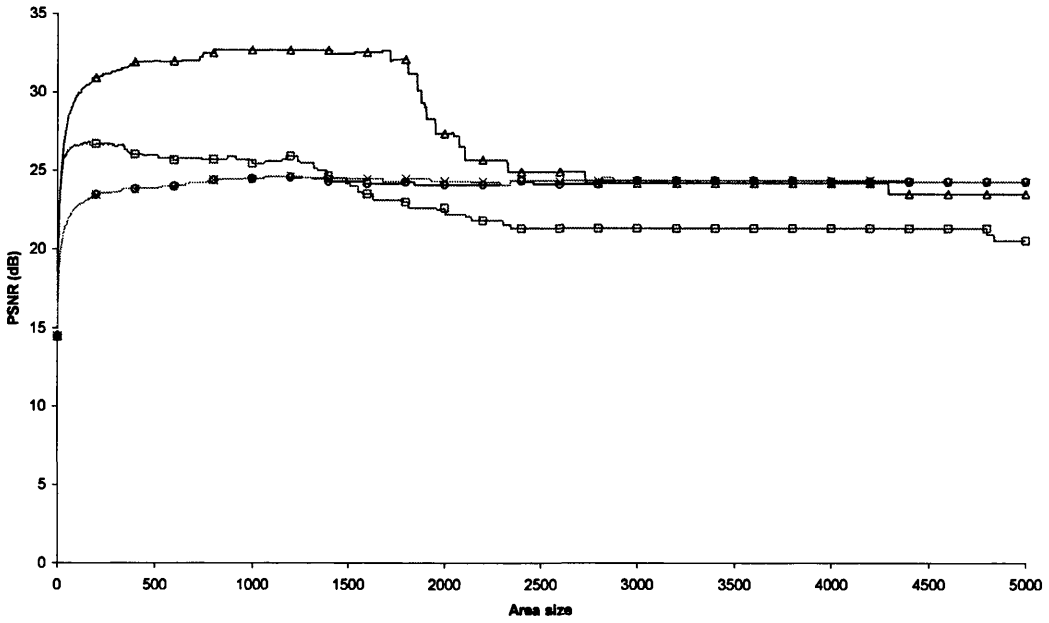
The power attribute (see Figures 7.14, 7.18, 7. 22 and 7.26) shows that the ASF and AF filter structures, when using 8nn connectivity are almost identical for every image used. All of the images show that the 4nn connectivity produces better results than the 8nn connectivity. However the 4nn connectivity also decreases in performance much faster. These graphs have shown conclusively that the 4nn and 8nn connectivity and the AF and ASF filter structures do not produce the same results. In addition this has shown how much variation in performance each attribute produces as the attribute value is increased. The general trend being that as the noise increases, the slower the degradation in performance is as the attribute value increases. Table 7.3 shows the optimum attribute values for the Simulated and Barbara. Results for Barbara 2, Boats and Goldhill images are given in Appendix B, Table 13.3. The results all show an improvement in the PSNR of the images. The simulated image shows that at high noise levels, 28dB, 21dB and 14dB, the 4nn connectivity and ASF filter structure achieve the optimum results. The Barbara, Barbara 2, Boats and Goldhill images all show that the optimum results are produced using the 4nn connectivity with the exception of the area constraint on the Barbara image with a noise level of 35dB. The area and contrast attributes perform best at high noise levels, 14dB and 21dB, using the ASF filter structure and 4nn connectivity, that is 100% of the images use this filter combination. However at lower noise levels, this swaps and the AF filter structure performs better with the exception of the contrast attribute and 28dB of noise. In this case, it is an even split between the two filter structures.

The simulated image using the area attribute (see Figures 7.11, 7.15, 7.19 and 7.23) shows a steady improvement in PSNR until it reaches a peak. For all combinations on all noise levels, except 14dB, the improvement quickly decreases around an area size of 1800. This is due to the fact that the circle in the simulated image has an area of 1877. Thus at this size, the circle will be removed completely. In general, the 4nn connectivity and ASF filter structure combination take much longer to reach their optimum, but in all cases achieves a better result. Using the AF filter structure generates the optimum much sooner, however the result is not as good as that achieved using the ASF filter structure. As the amount of noise increases, the difference between the performances of the two filter structures using 4nn connectivity becomes more apparent. In addition 8nn connectivity generally takes longer to reach

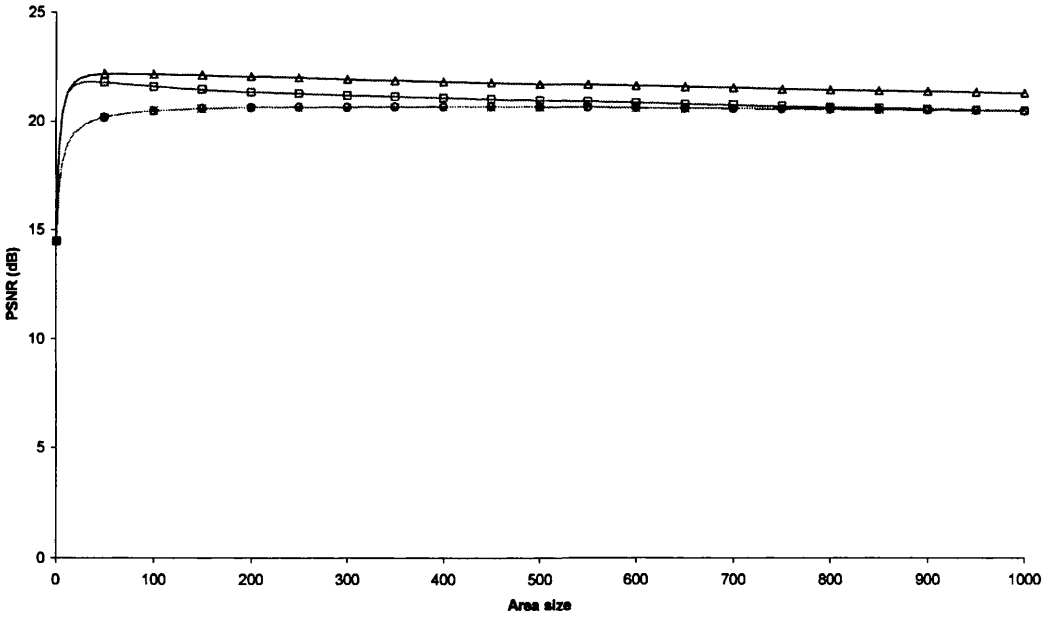
the optimum value than the corresponding filter structure using 4nn connectivity. This relationship holds for all of the attributes. The results for the Barbara, Barbara 2, Boats and Goldhill images using the area attribute all show the same patterns. With a high amount of noise (14dB), the improvement increases slowly and then decreases slowly. However, as the noise level decreases, the optimum value is reached much quicker and decreases much more rapidly. The difference between the 4nn connectivity and ASF filter structure combination and the other combinations becomes less apparent as the noise levels decrease. In addition the Barbara, Barbara 2, Boats and Goldhill images confirm that the 4nn and ASF filter structure combination performs better than most other combinations, regardless of the noise level.

The contrast attribute (see Figures 7.12, 7.16, 7.20 and 7.24) shows that unlike the area attribute, the ASF and AF filter structures have a similar performance regardless of the connectivity used. For example using the simulated image, the ASF filter structure shows that the 4nn connectivity works better than the 8nn connectivity but that their performance is a close match with the 8nn connectivity only being about 2dB lower. However for the AF filter structure, the results are almost identical regardless of the connectivity. The simulated results also show a peak using the AF filter structure and 4nn connectivity, which is at a high value for a high amount of noise (200 for example with 14dB of noise) and decreases as the noise is reduced (20 for 35dB of noise). All of the results for the contrast operator show that the ASF filter structures, once they have reached their optimum value decrease in performance relatively slowly when compared to the AF filter structure, which tends to decrease rapidly. Unlike the simulated image, the Barbara, Barbara 2, Boats and Goldhill images show the ASF filter structure and 4nn connectivity producing better results than any other combination.

Unlike the area and contrast attributes, the volume attribute (see Figures 7.13, 7.17, 7.21 and 7.25) shows clearly that there is a difference between the different combinations of filter structure and connectivity. All images show that once the optimum is reached, that there is a slow degradation in performance. At low noise levels (35dB), the degradation is fast compared to higher noise levels. However the degradation in performance does not decrease as fast as the contrast does. In addition, like the contrast attribute, the two ASF combinations appear to be linked, as do the two AF combinations. The volume attribute is the only one that remains constant. It produces the optimum filtered image using 4nn and the ASF filter structure regardless of the noise level. The power attribute, like area and contrast produces the best output using 4nn connectivity and the ASF filter structure for all images with the exception of the Barbara 2 image. That is 75% of the time this filter combination provides the optimum output. At the highest level of noise, 14dB, the area attribute also produces the best results, whilst at all other levels the power attribute produces the optimum image.



a) Simulated image (128 x 128, 8bit greyscale).

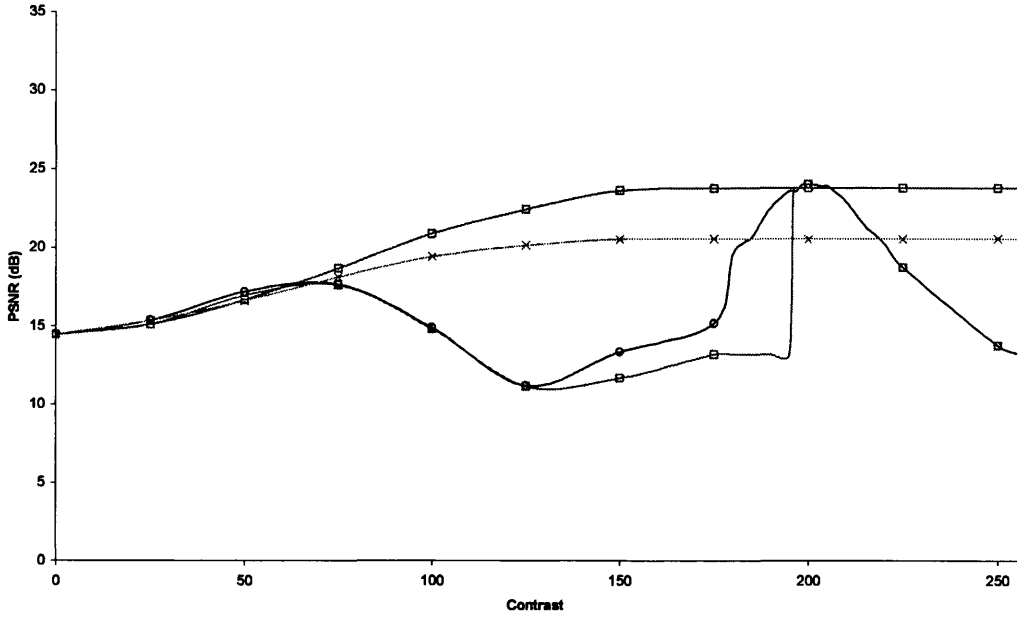


b) Barbara image (720 x 576, 8bit greyscale).

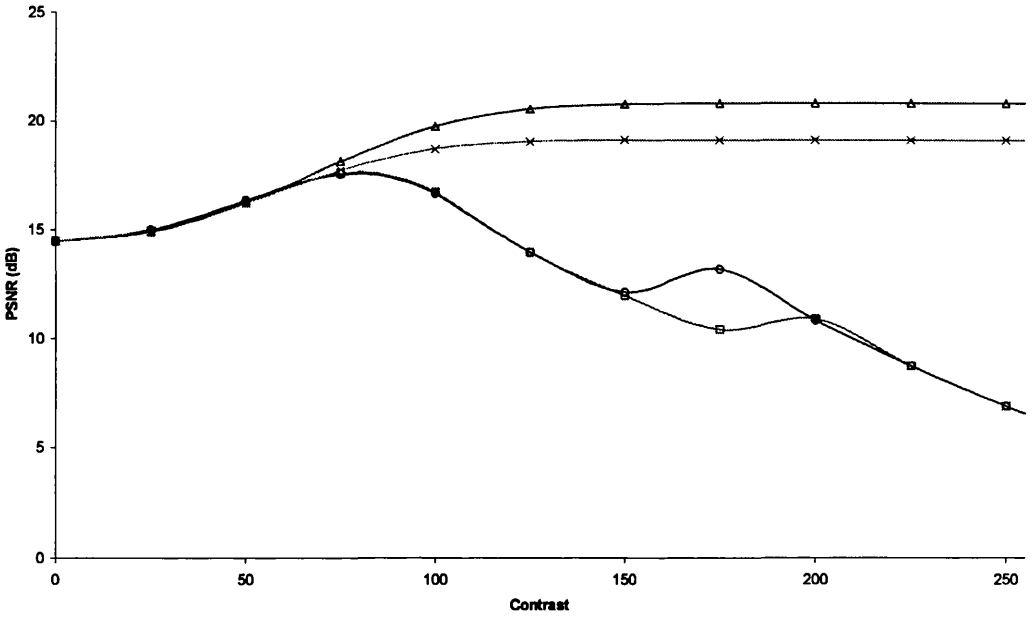
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.11: Filter performance with respect to an increasing area size using an image corrupted with 14dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

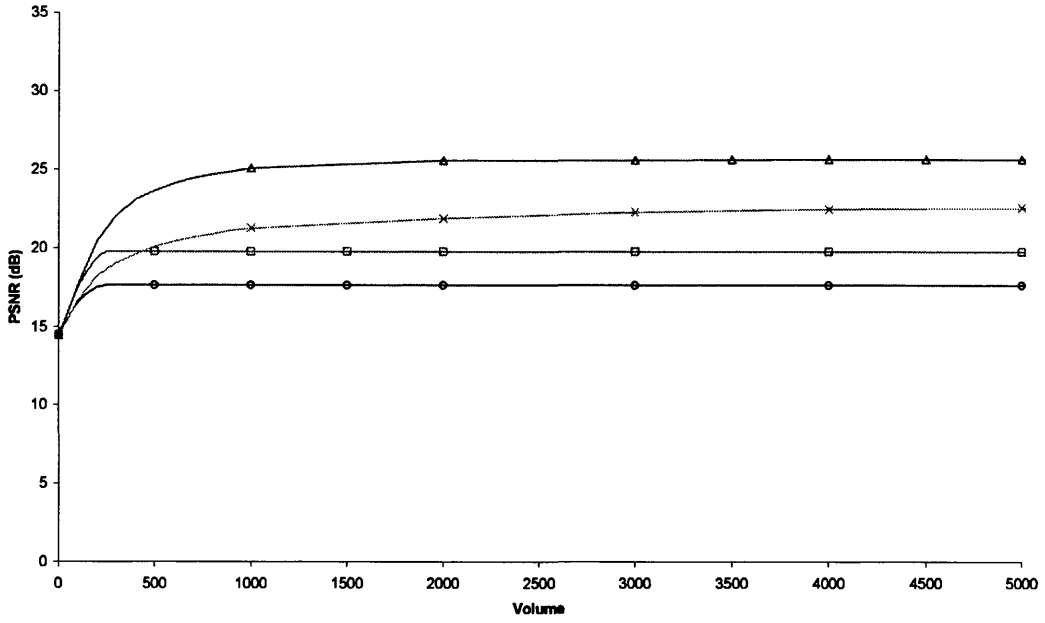


b) Barbara image (720 x 576, 8bit greyscale).

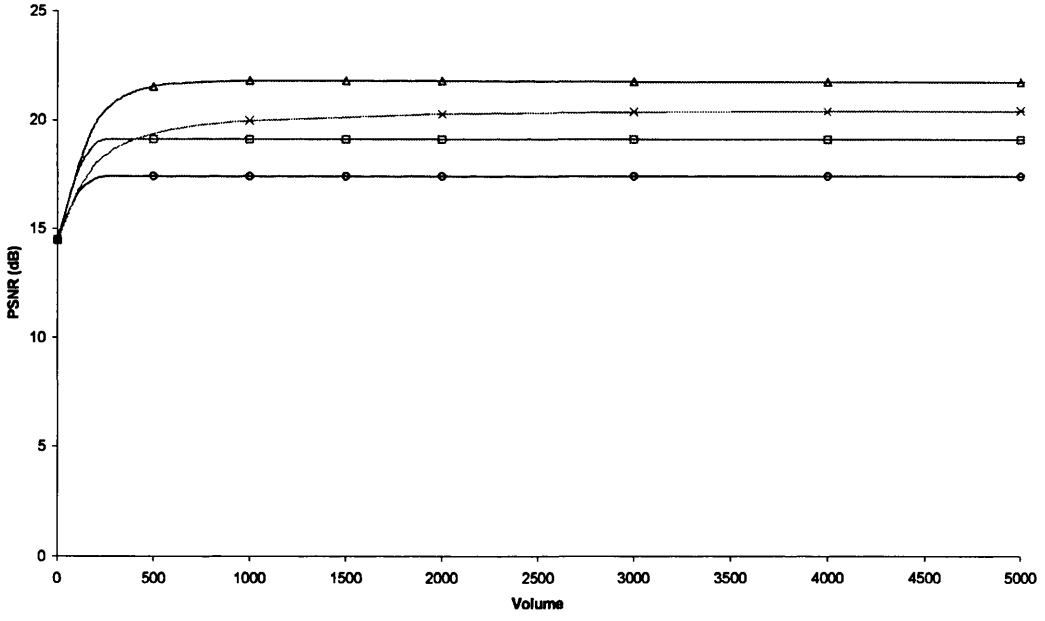
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.12: Filter performance with respect to an increasing contrast attribute using an image corrupted with 14dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

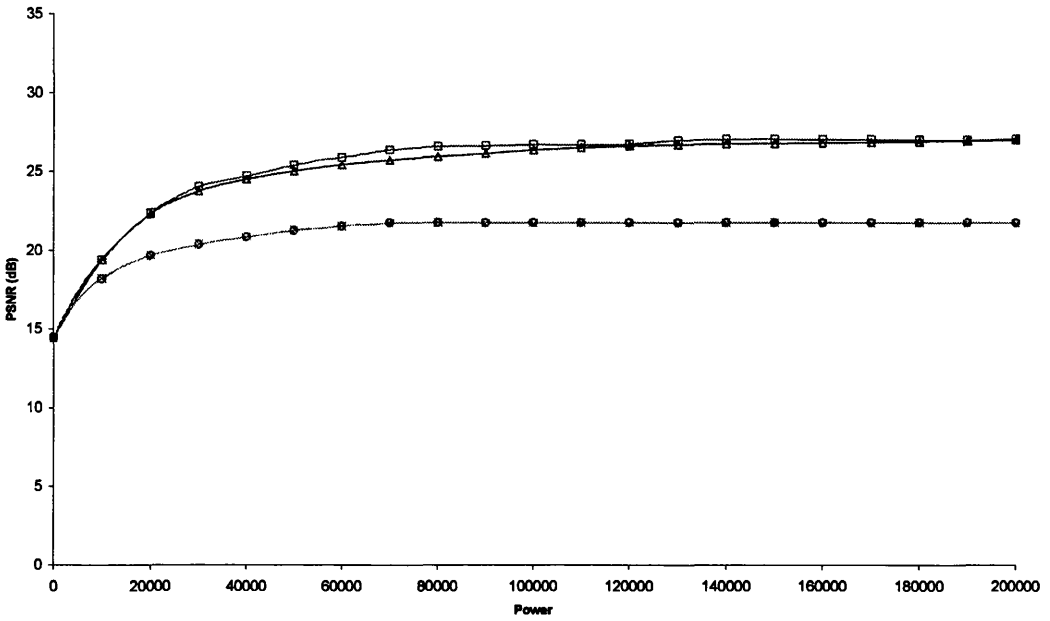


b) Barbara image (720 x 576, 8bit greyscale).

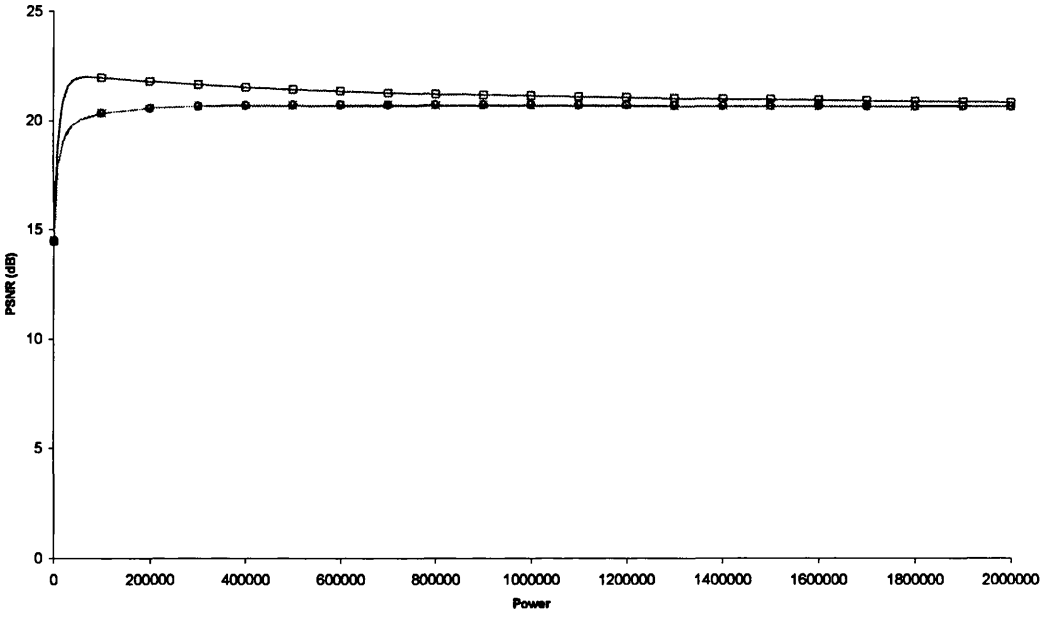
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.13: Filter performance with respect to an increasing volume attribute using an image corrupted with 14dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

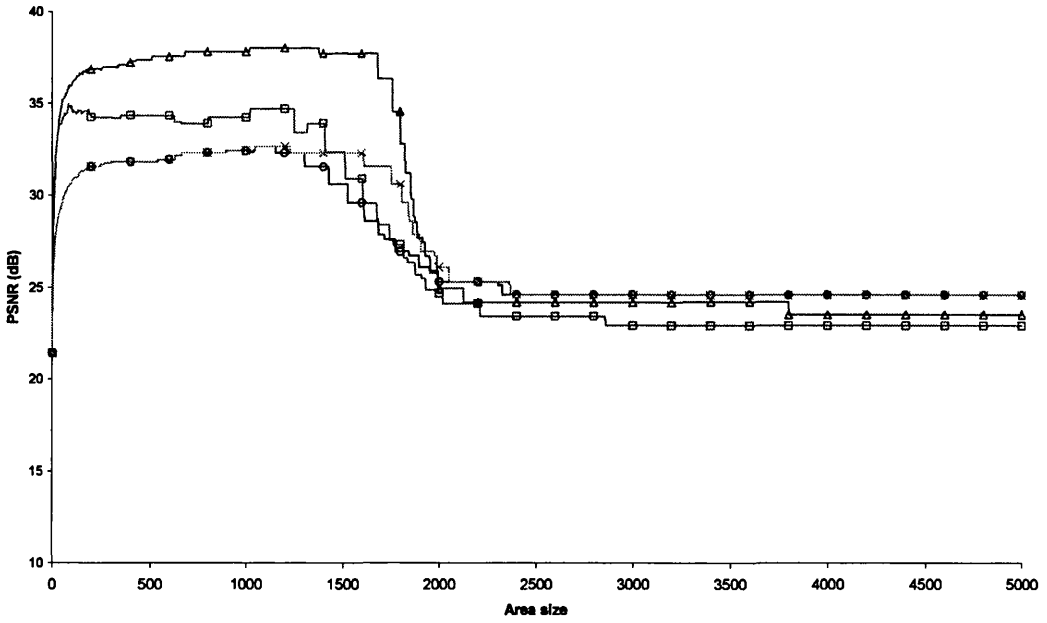


b) Barbara image (720 x 576, 8bit greyscale).

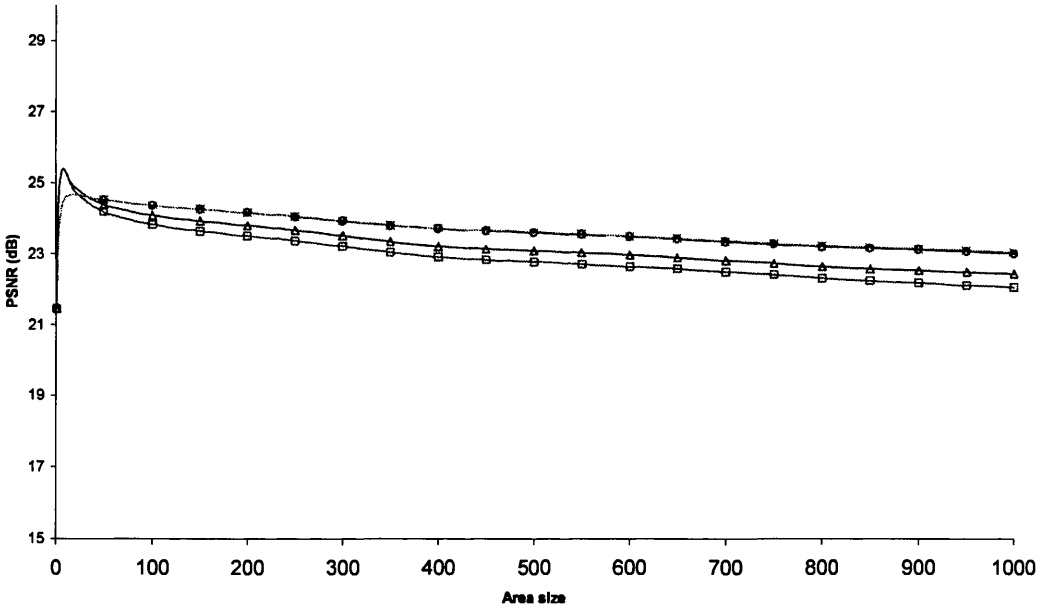
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.14: Filter performance with respect to an increasing power attribute using an image corrupted with 14dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

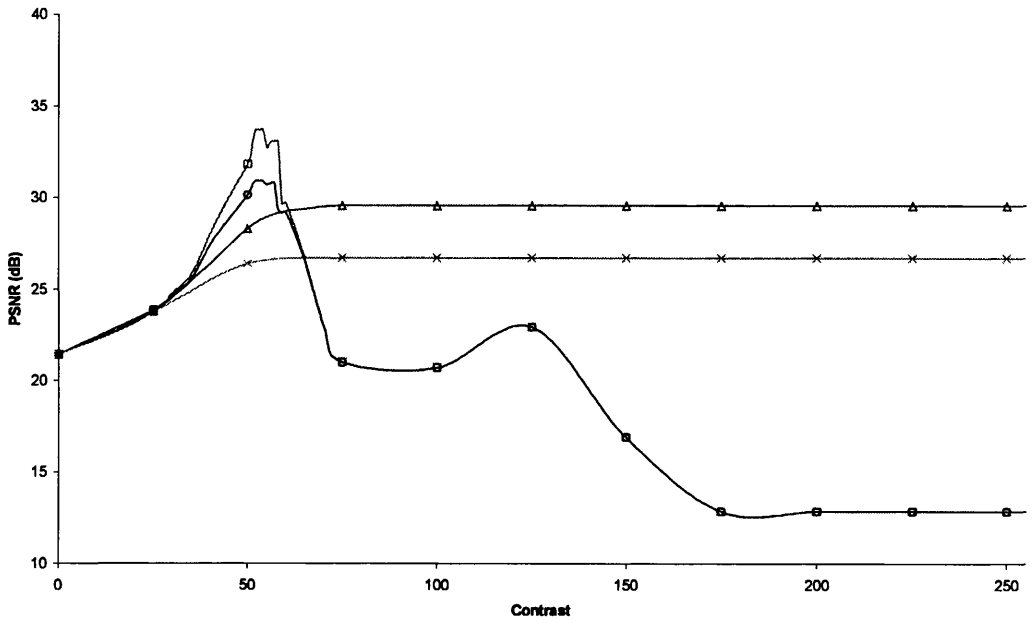


b) Barbara image (720 x 576, 8bit greyscale).

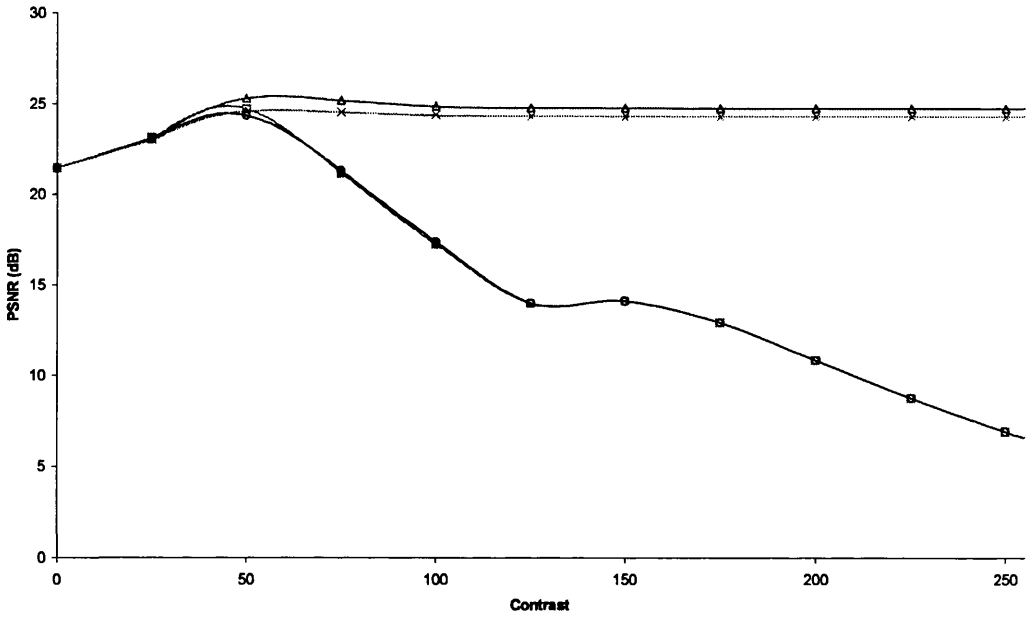
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.15: Filter performance with respect to an increasing area attribute using an image corrupted with 21dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

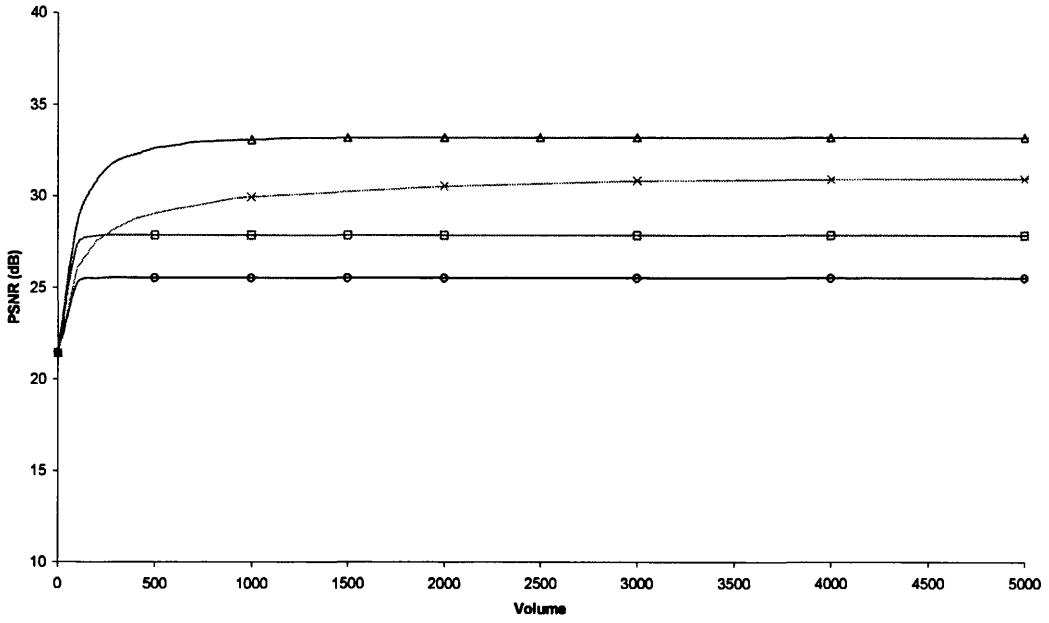


b) Barbara image (720 x 576, 8bit greyscale).

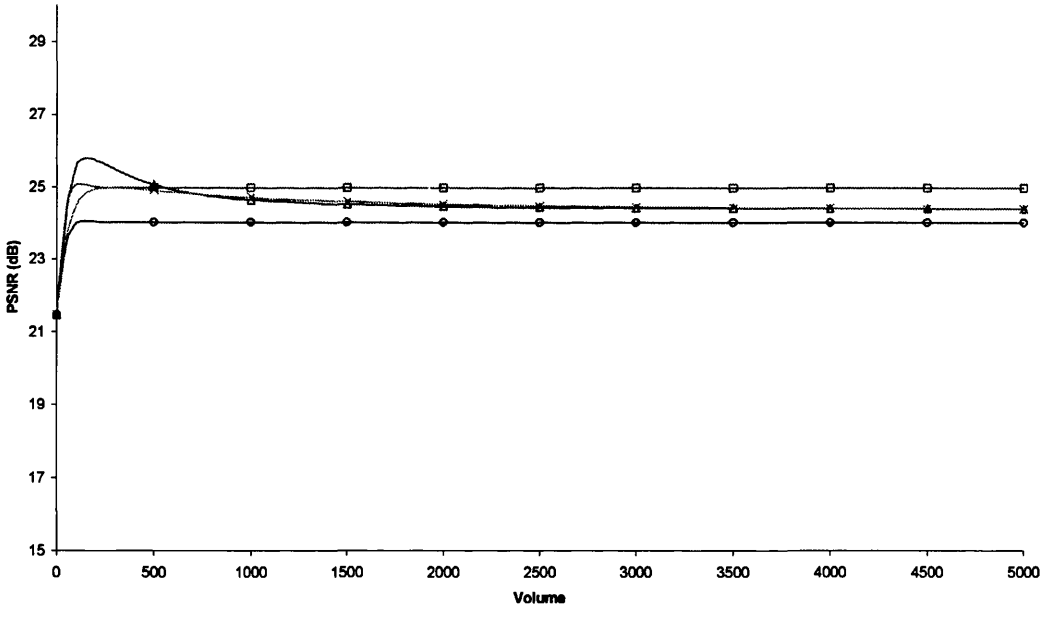
Key: 4nmAF 4nmASF 8nmAF 8nmASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.16: Filter performance with respect to an increasing contrast attribute using an image corrupted with 21dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

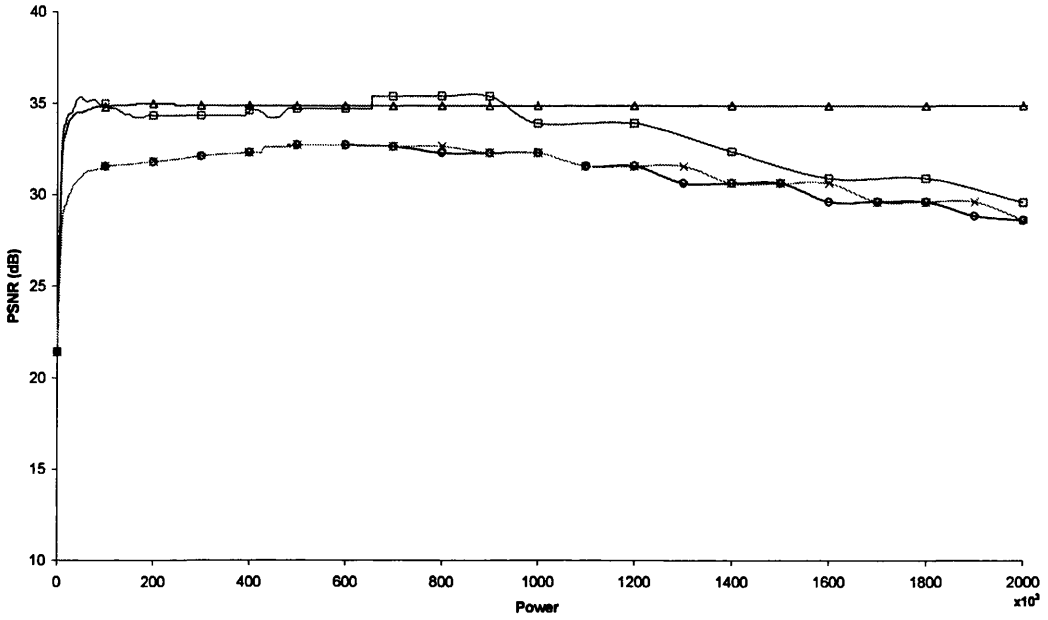


b) Barbara image (720 x 576, 8bit greyscale).

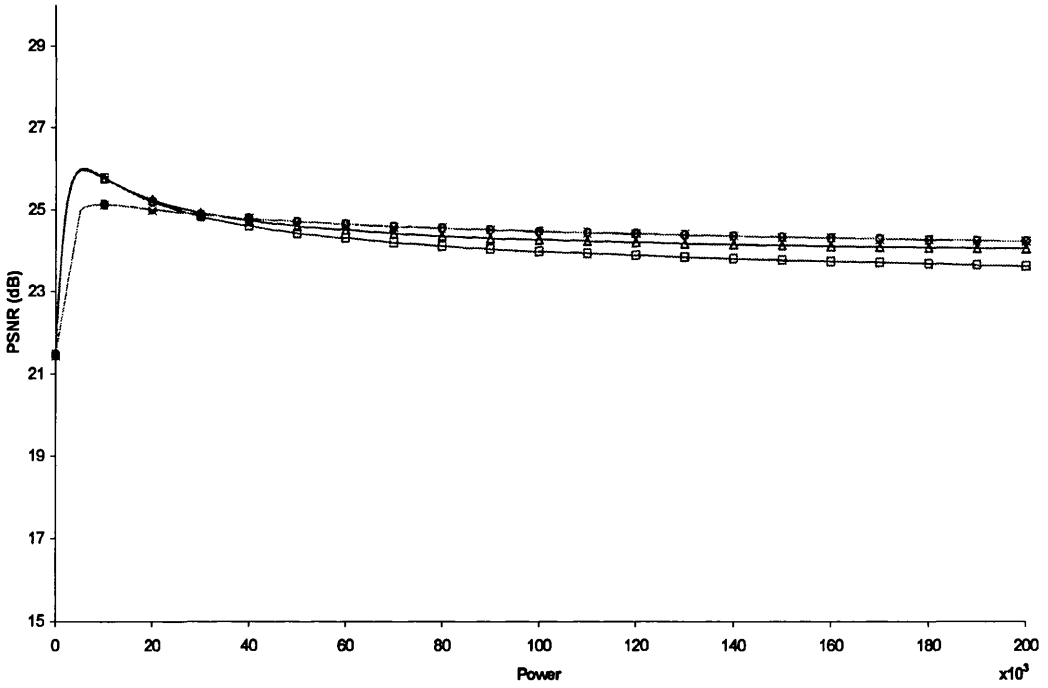
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.17: Filter performance with respect to an increasing volume attribute using an image corrupted with 21dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

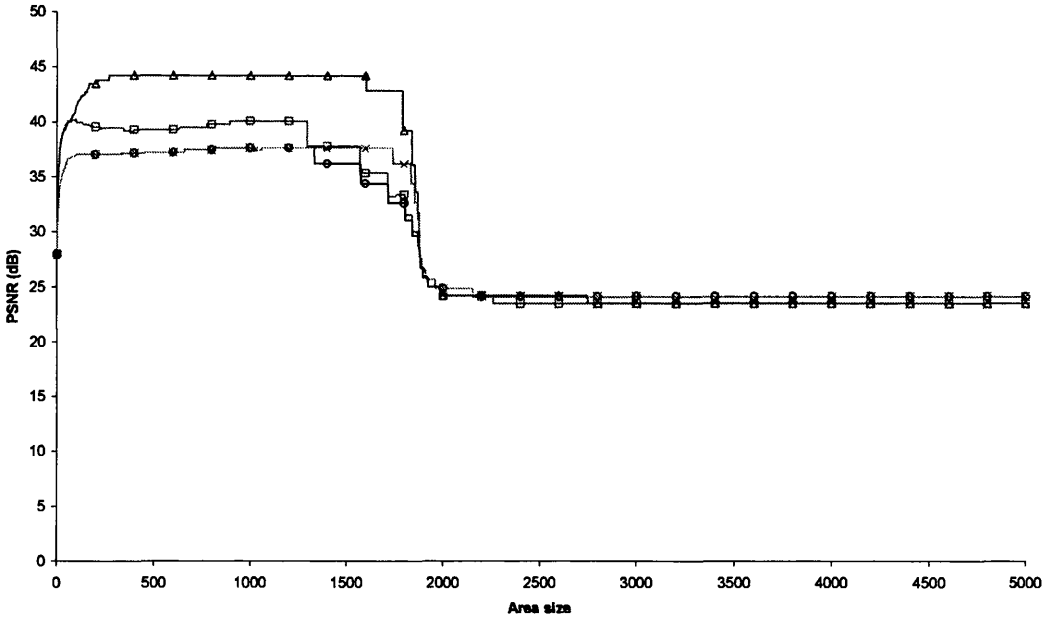


b) Barbara image (720 x 576, 8bit greyscale).

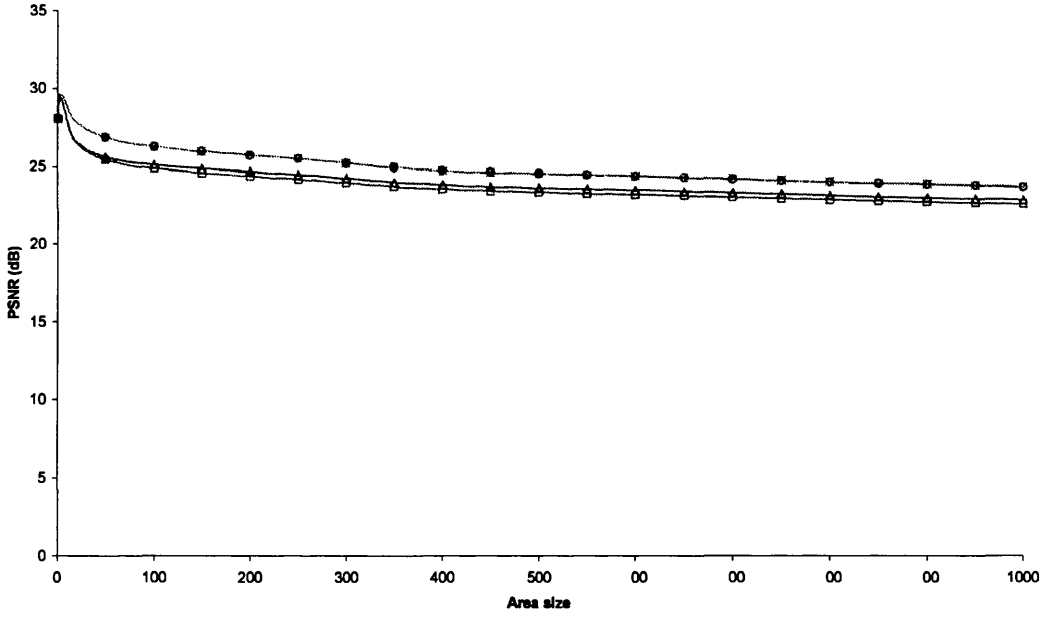
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.18: Filter performance with respect to an increasing power attribute using an image corrupted with 21dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

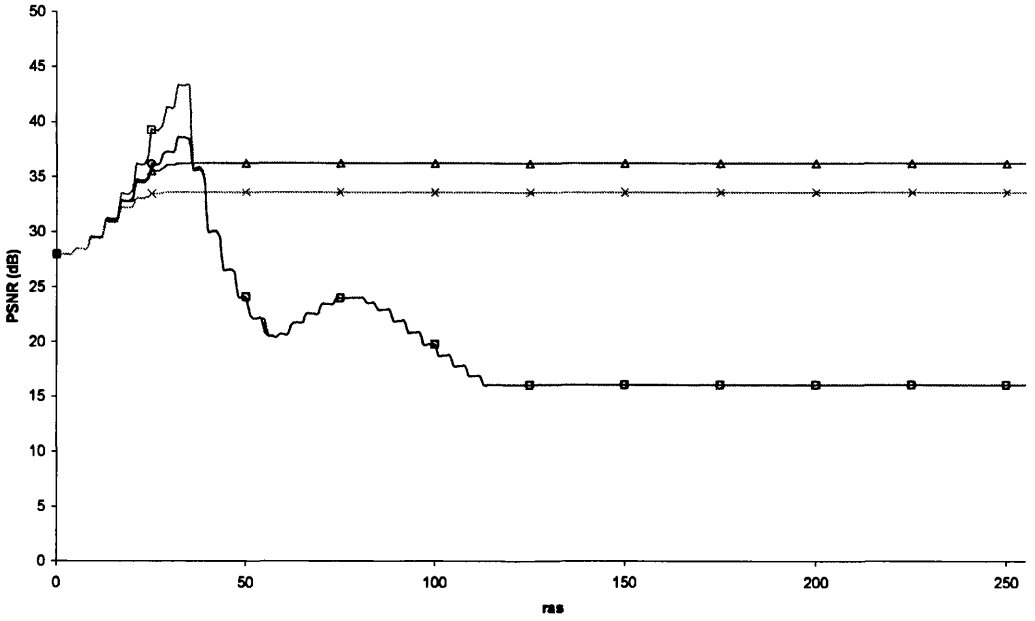


b) Barbara image (720 x 576, 8bit greyscale).

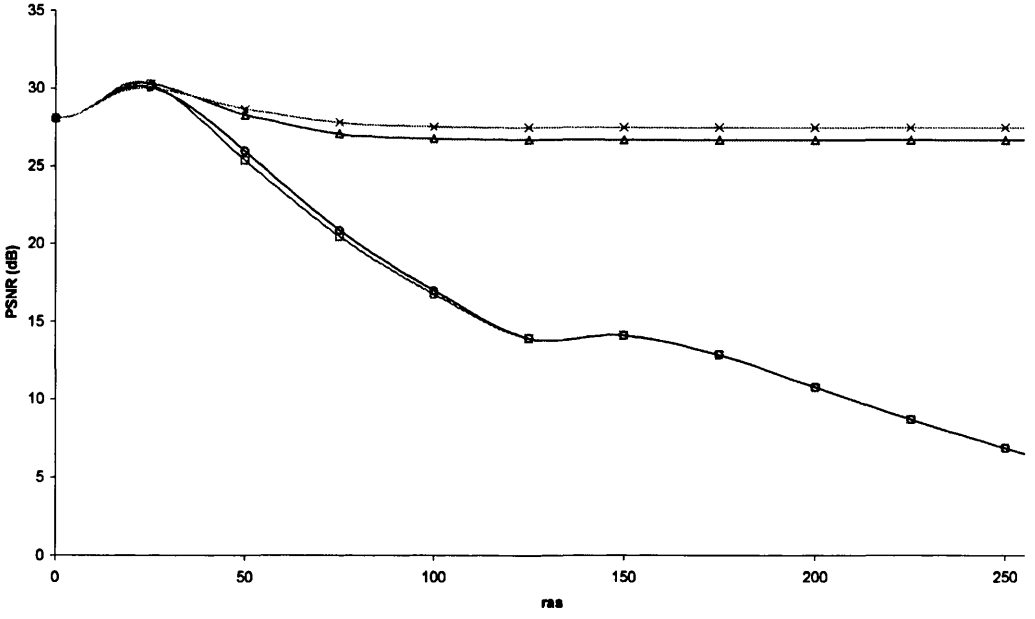
Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.19: Filter performance with respect to an increasing area attribute using an image corrupted with 28dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

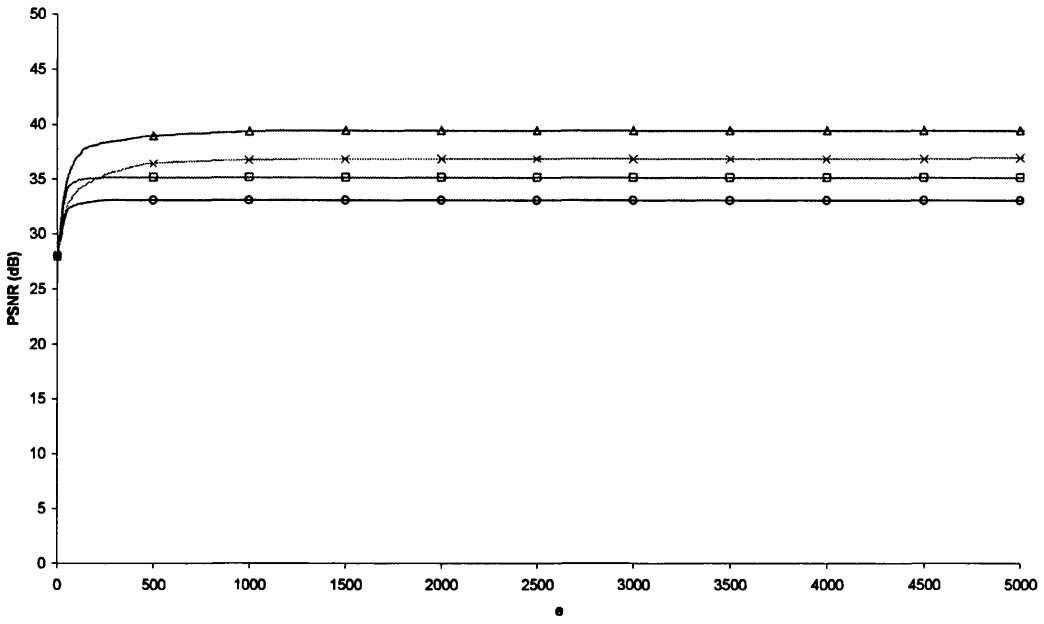


b) Barbara image (720 x 576, 8bit greyscale).

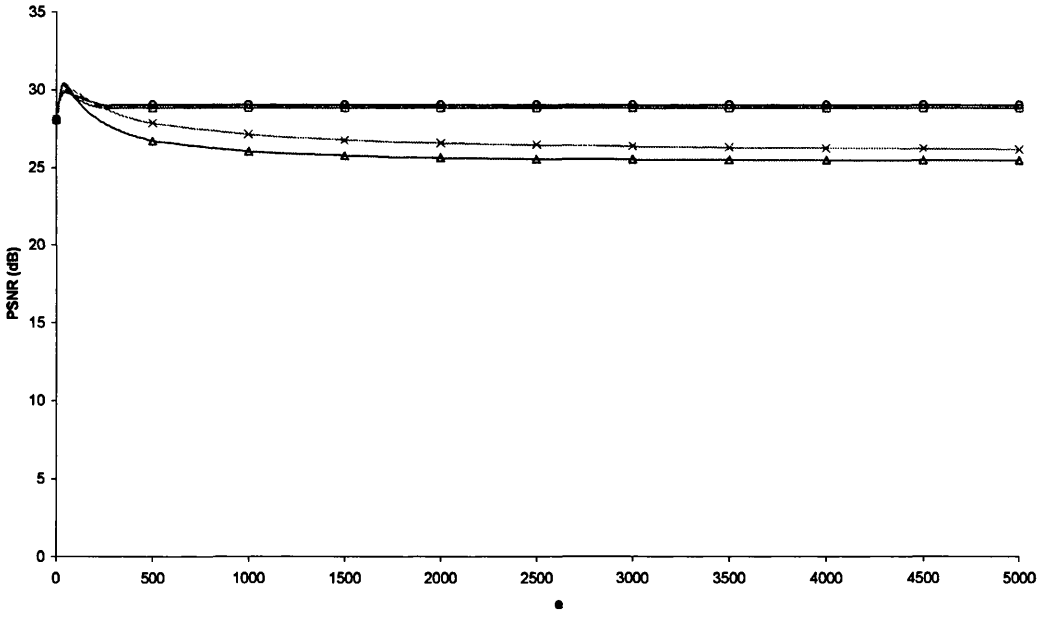
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.20: Filter performance with respect to an increasing contrast attribute using an image corrupted with 28dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

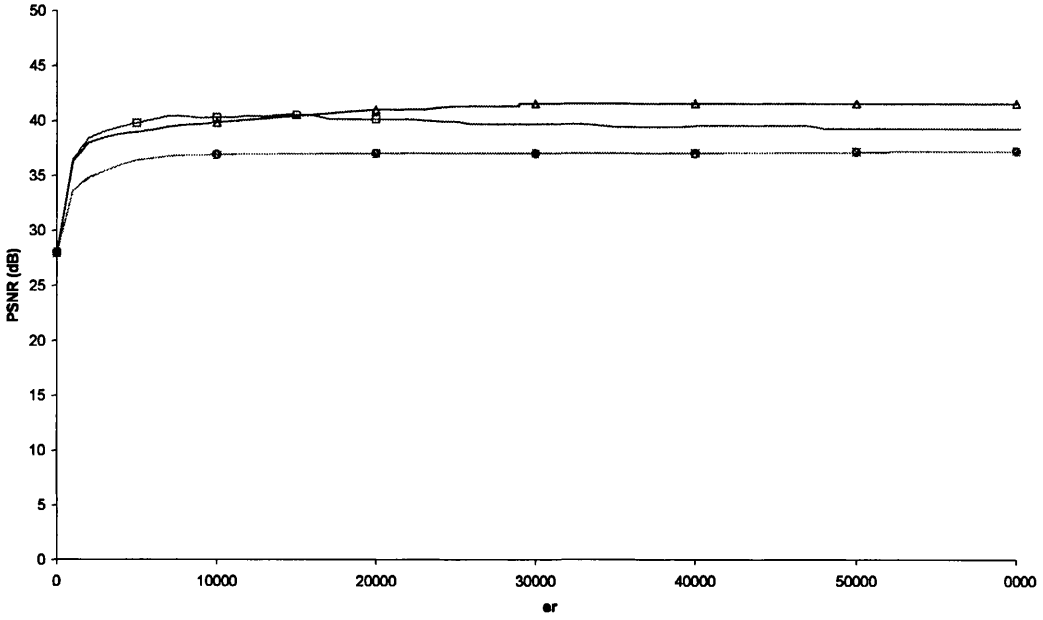


b) Barbara image (720 x 576, 8bit greyscale).

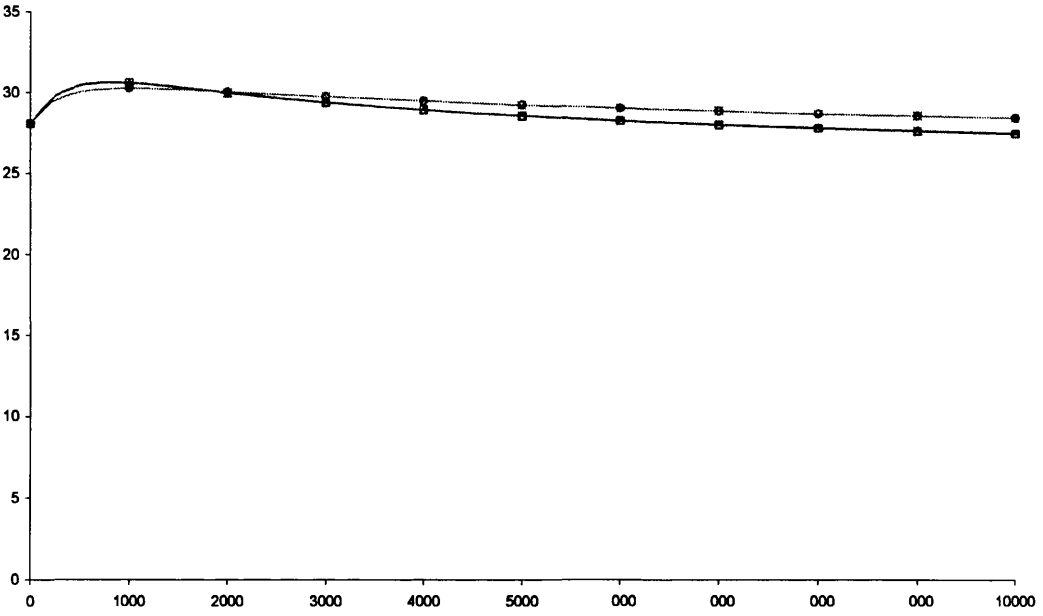
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.21: Filter performance with respect to an increasing volume attribute using an image corrupted with 28dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

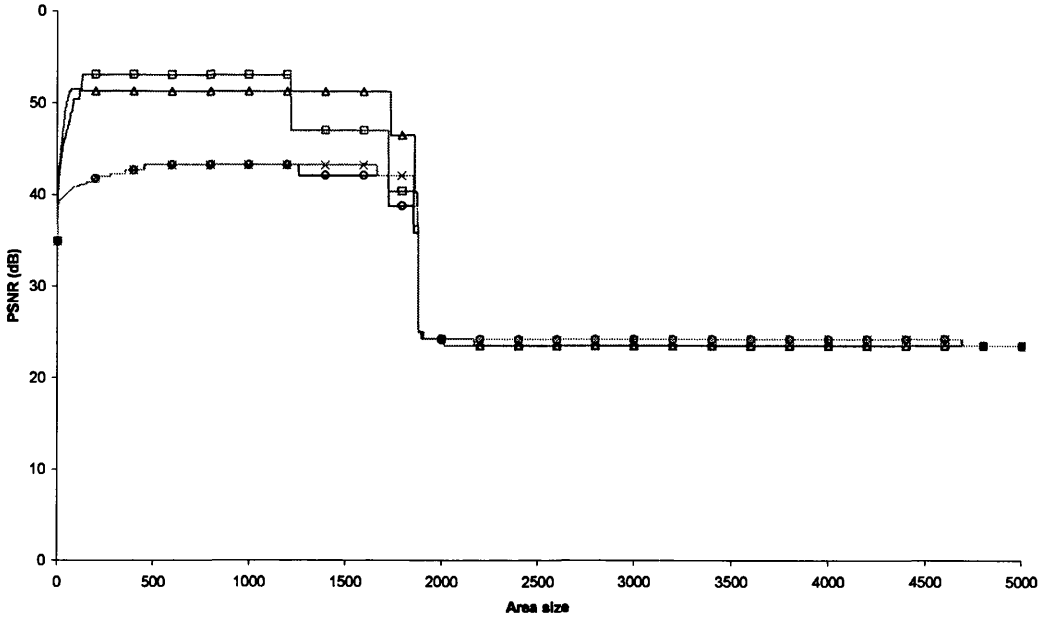


b) Barbara image (720 x 576, 8bit greyscale).

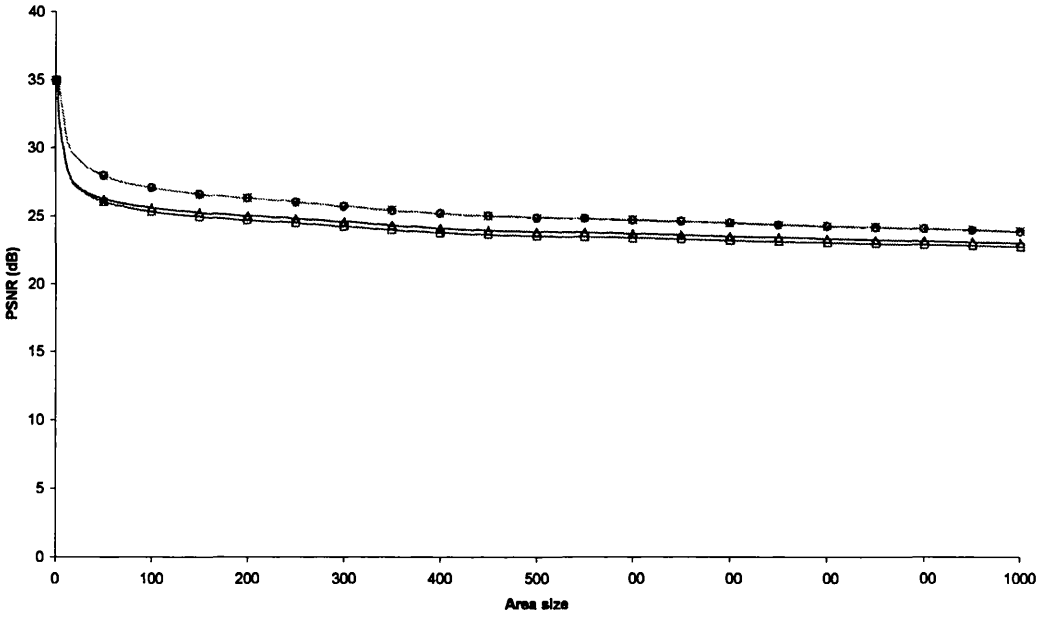
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs un readable.

Figure 7.22: Filter performance with respect to an increasing power attribute using an image corrupted with 28dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

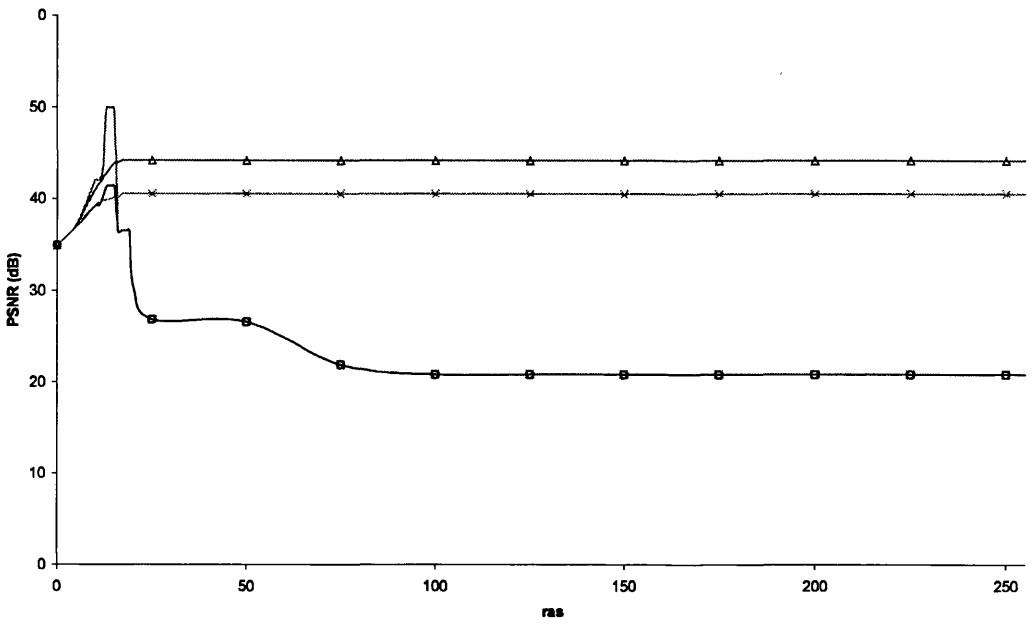


b) Barbara image (720 x 576, 8bit greyscale).

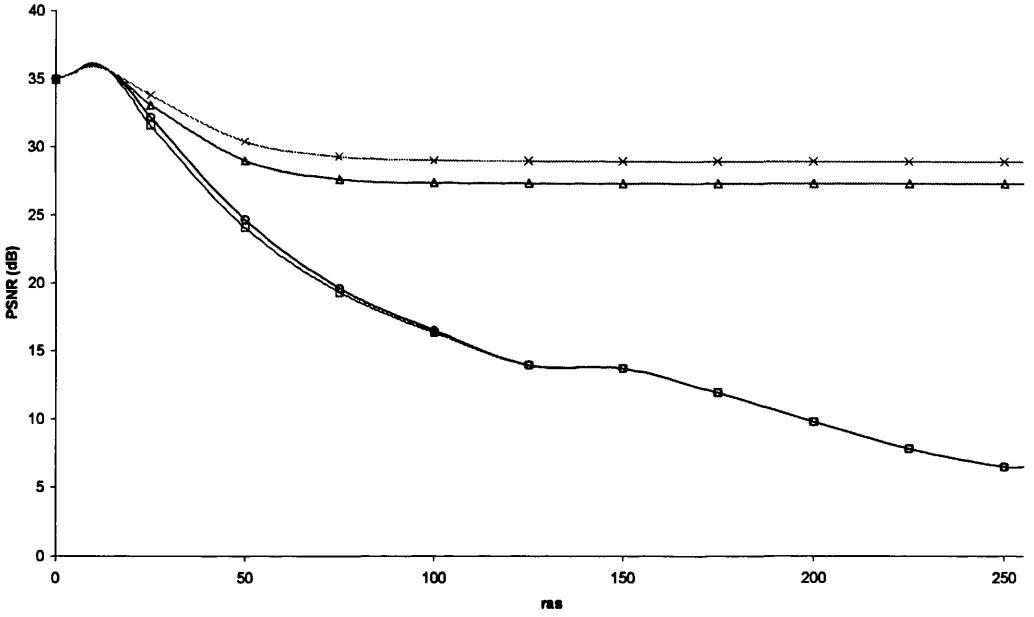
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.23: Filter performance with respect to an increasing area attribute using an image corrupted with 35dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

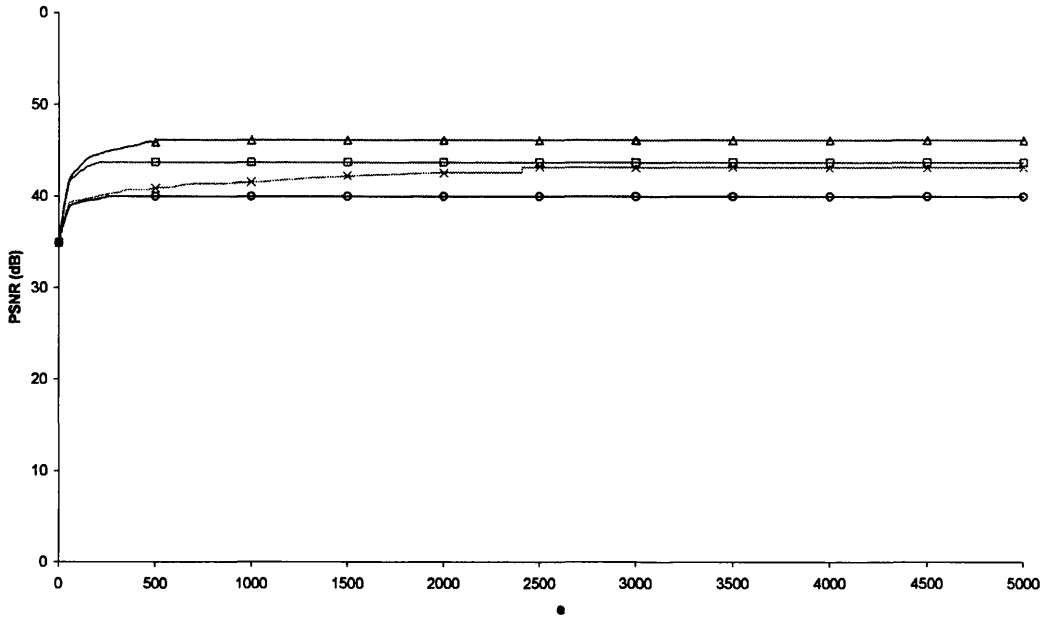


b) Barbara image (720 x 576, 8bit greyscale).

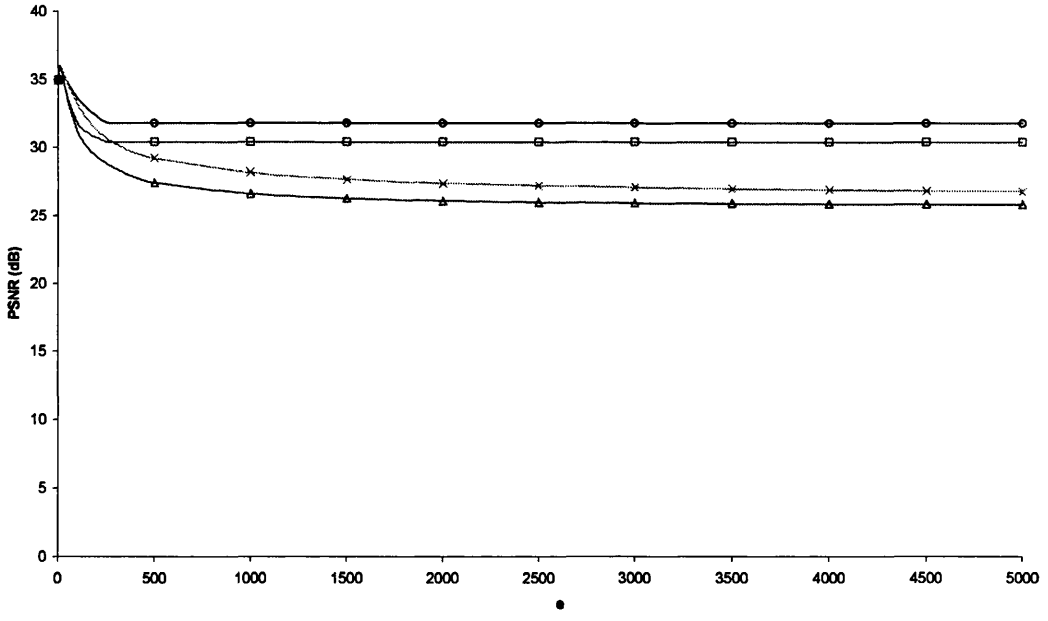
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.24: Filter performance with respect to an increasing contrast attribute using an image corrupted with 35dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).

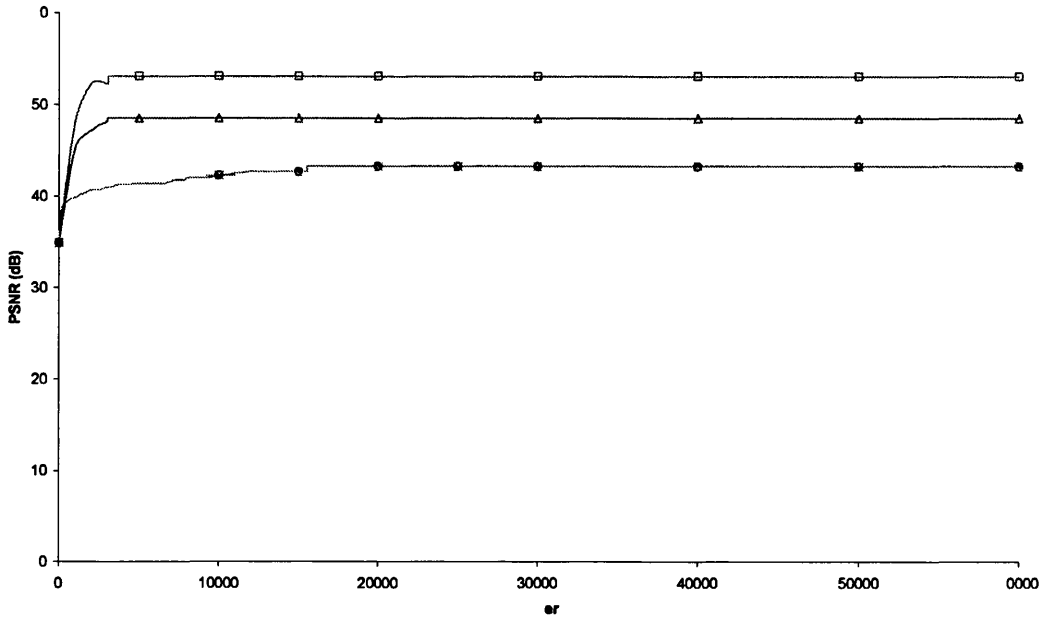


b) Barbara image (720 x 576, 8bit greyscale).

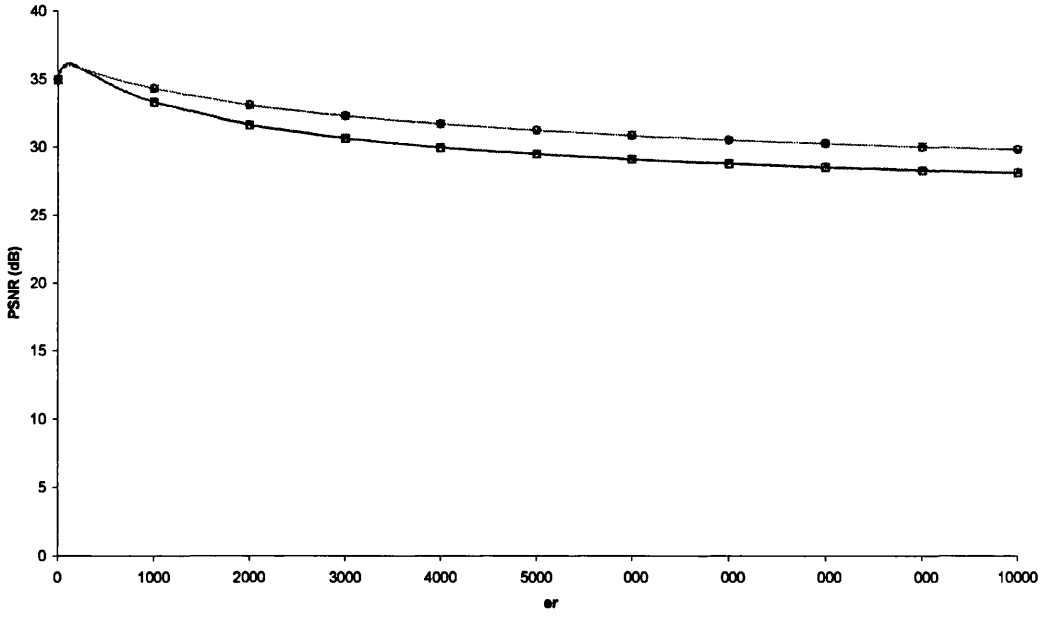
Key: 4nnAF 4nnASF 8nnAF 8nnASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.25: Filter performance with respect to an increasing volume attribute using an image corrupted with 35dB of noise.



a) Simulated image (128 x 128, 8bit greyscale).



b) Barbara image (720 x 576, 8bit greyscale).

Key: 4mAF 4mASF 8mAF 8mASF

Note that the markers are shown to identify the data and do not represent where measurements were made, as this would have made the graphs unreadable.

Figure 7.26: Filter performance with respect to an increasing power attribute using an image corrupted with 35dB of noise.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 14.48dB	Area	35	21.82	56	**22.19**	438	20.68	441	20.68
	Contrast	80	17.68	181	20.86	80	17.64	237	19.15
	Volume	256	19.13	1123	21.81	256	17.45	8485	20.45
	Power	71210	22.02	70713	22.03	834790	20.73	839600	20.73
Sim2 14.47dB	Area	157	26.85	808	**32.76**	1064	24.67	1064	24.67
	Contrast	199	24.06	180	23.82	199	24.06	180	20.58
	Volume	256	19.81	3761	25.68	256	17.68	8838	22.85
	Power	257623	27.24	71089	25.81	70978	21.78	71005	21.78
Barbara 21.46dB	Area	8	25.39	8	25.40	18	24.67	18	24.67
	Contrast	45	24.88	58	25.41	45	24.47	59	24.64
	Volume	113	25.09	156	25.80	146	24.06	267	24.99
	Power	5780	**26.01**	5778	25.98	9550	25.14	9550	25.14
Sim2 21.44dB	Area	84	34.97	1018	**38.08**	1046	32.68	1046	32.68
	Contrast	53	33.70	74	29.59	52	30.92	76	26.72
	Volume	255	27.88	1682	33.24	245	25.54	7220	31.15
	Power	654555	35.43	164823	35.01	479743	32.75	20870	32.75
Barbara 28.09dB	Area	2	29.65	2	29.65	4	29.59	4	29.59
	Contrast	22	30.45	23	30.36	22	30.16	23	30.04
	Volume	37	30.31	39	30.46	44	29.85	55	30.11
	Power	793	**30.66**	793	30.63	973	30.26	973	30.26
Sim2 28.01dB	Area	95	40.23	269	**44.26**	1060	37.68	1060	37.68
	Contrast	35	43.45	36	36.28	32	38.62	36	33.64
	Volume	254	35.21	1087	39.48	253	33.13	4749	37.00
	Power	14451	40.62	28985	41.60	52028	37.27	52028	37.27
Barbara 34.96dB	Area	1	34.96	1	34.96	2	35.31	2	35.31
	Contrast	10	36.17	10	36.10	10	36.01	10	35.94
	Volume	12	36.03	12	36.04	13	35.82	14	35.85
	Power	116	**36.19**	116	36.16	136	35.99	136	35.99
Sim2 34.96dB	Area	130	53.13	115	**51.33**	455	43.32	456	43.32
	Contrast	13	49.93	17	44.23	13	41.40	17	40.60
	Volume	234	43.75	510	46.15	253	40.02	2409	43.19
	Power	3066	53.13	3010	48.55	15553	43.32	15562	43.32

Table 7.3: The optimum attribute values for noise removal and their performance gain in PSNR.

The shaded values show the best outputs and the outlined show the worst.

Figure 7.27 shows the original uncorrupted Barbara image and two images corrupted with 14dB and 35dB of AWGN (see section 5.1.1). The filtered outputs are shown in Figure 7.28 using the 14dB corrupted version of the Barbara image. The first image, Figure 7.28a, shows the optimum morphological filter, which uses the area attribute, 4nn connectivity and the ASF filter structure (see section 4.6.2). This produces a PSNR of 22.19dB, an improvement of 7.71dB, where as the optimum Gaussian filter (see section 7.1.1 and Table 7.4) gives an increase of 8.56dB, but tends to blur the image slightly. The last image, Figure 7.28c shows the output of the worst performing filter, which uses the volume attribute, 8nn connectivity and the AF filter structure giving a PSNR of 17.45dB. Visually, the noise is very much visible in this image, although some amount of it has been reduced. The second image, Figure 7.28b, shows a result that is roughly in-between the optimum and the worst. It uses the power attribute, 8nn connectivity and the ASF filter structure, which produces a PSNR of 20.73dB. Figure 7.29 shows the output of filtering for the Barbara image using 35dB of noise. The first image, Figure 7.29a, shows the optimum filter output, which uses the power attribute, 4nn connectivity and the AF filter structure producing a PSNR of 36.19dB. In contrast, the optimum Gaussian method (see section 7.1.1 and Table 7.4) produced a PSNR of 35.52dB. The last image, Figure 7.29c shows the worst output. This uses the area attribute, 4nn connectivity, and both the AF and ASF filter structures. As the attribute value is only 1 and the area attribute is used for both the AF and ASF filter structures, no filtering is actually performed, thus the output of the filter is the same as the input. Hence no noise is actually removed. Figure 7.29b, the second image, shows the result of using 8nn connectivity, the AF filter structure and the volume attribute. The PSNR achieved using this is 35.82dB and lays between the optimum and worst filters. From the figures and the tables, several conclusions can be drawn; the most important of these is demonstrated best by the simulated image, which shows a distinct difference between the filter structures and connectivity. In addition, the 4nn connectivity produces the optimum output for the majority of the four test images (Barbara, Barbara 2, Boats and Goldhill). The new power attribute has been shown to perform significantly better than all other attributes, with the exception of the area attribute on 14dB of noise. This has shown conclusively that when designing morphological filters, the filter structure, connectivity and attribute used should be considered carefully.

a) Original uncorrupted Barbara image.

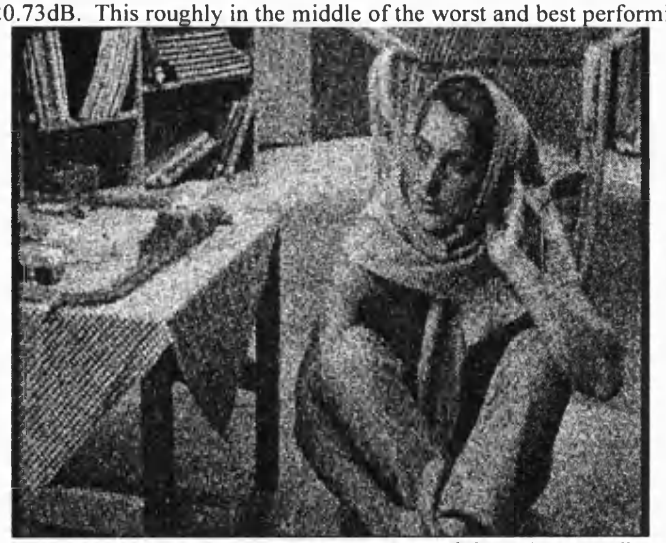
b) 14dB of AWGN added to the Barbara image.

c) 35dB of AWGN added to the Barbara image.

Figure 7.27: Corrupted input images of Barbara (720 x 576. 8bit greyscale) with AWGN of 14dB and 35dB that are used as the input to the morphological filter.

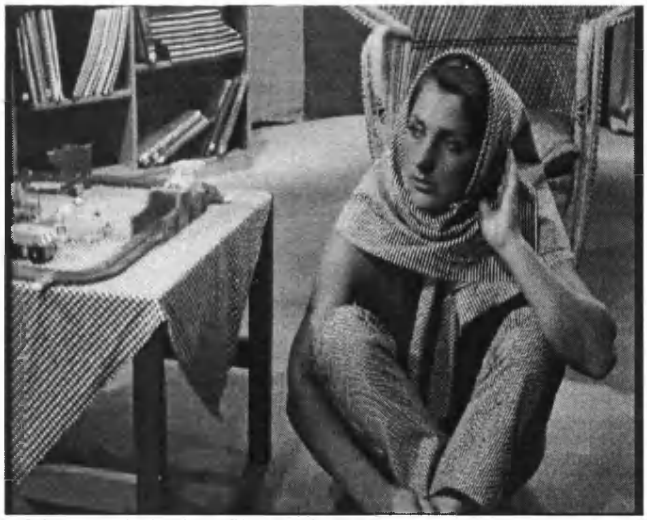
a) The best output of the morphological filter (using 4nn, AF and the power attribute), which gives a PSNR of 22.19dB.

b) Output of the morphological filter using 8nn, the ASF filter structure and the power attribute, which gives a PSNR of 20.73dB. This roughly in the middle of the worst and best performing combinations.



c) Output of the worst morphological filter (using 8nn, AF and the volume attribute), which gives a PSNR of 17.45dB.

Figure 7.28: Morphological filter output using the Barbara image (720 x 576. 8bit greyscale) with 14dB of noise added.



a) The best output of the morphological filter (using 4nn, AF and the power attribute), which gives a PSNR of 36.19dB.

b) Output of the morphological filter using 8nn, the AF filter structure and the volume attribute, which gives a PSNR of 35.82dB. This roughly in the middle of the worst and best performing combinations.

c) Output of the worst morphological filter (using 4nn, AF and the area attribute), which gives a PSNR of 34.96dB (i.e. it is identical to the noisy input image).

Figure 7.29: Morphological filter output using the Barbara image (720 x 576. 8bit greyscale) with 35dB of noise added.

A comparison between the best morphological results (see Table 7.3) and the results obtained using the Gaussian filtering method (see Table 7.1) is given in Table 7.4. This shows that the Gaussian filter is better when the image contains a large amount of noise as it comes out the best for 60% and 80% using 14dB and 21dB of noise respectively. For a lower amount of noise, 28dB and 35dB, the morphological filtering method comes out best 80% of the time for both cases. Despite this, the two results are very similar, deviating by only a small amount. For example using the Barbara image, there is only a 0.85dB difference with 14dB of noise present and for 35dB of noise the difference is only 0.67dB. There are only a few large differences, with the exception of the Simulated image, where the morphological filter always performs better than the Gaussian. This is due to the fact that the Gaussian will blur the edges and introduces new image values, whereas the morphological method will not. Hence the best filter to use will not only depend upon the amount of noise present, but also the image content. This comparison has shown that although the Gaussian is better at filtering images with a large amount of noise, the morphological filtering is very similar in performance and is usually better with simple images, the Simulated image for example, and at low noise levels.

7.1.2.1 Compression Results

The attribute filters have shown to be able to remove noise relatively well. However, the idea in preprocessing is to improve the compressibility of the data. Thus by removing the noise from the image, the resultant image should compress better than the noisy image. Ideally the preprocessed image should be identical to the original noise free image. The optimum preprocessor settings found in section 7.1 are used to filter the images using each of the combinations of filter structure, connectivity and attribute. The resultant images are then compressed and decompressed using JPEG (see Table 7.5) and JPEG 2000 (see Table 7.6) to a compression ratio of 20:1 (0.4bpp). Results for the remainder of the test images are shown in Appendix B, Tables 13.5 – 13.8 (see section 13.1.3). This shows that the output of the CODEC using the filtered images is a closer match to the original noise free image after being compressed than the noisy image is.

Table 7.5 shows that by just compressing and uncompressing the original, using JPEG, Barbara image results in a PSNR of 45.02dB. However, when the noisy version is compressed, it results in a PSNR of only 18.59dB showing that the JPEG CODEC simply cannot compress an image corrupted with noise efficiently. The CODEC has however performed some filtering internally as the compressed output has a better PSNR, 18.59dB, than the corrupted input image, 14.48dB. This is due to the way in which the DCT and quantiser work. Using the area attribute in combination with the ASF filter structure and 4nn connectivity gave the optimum result for the noise reduction as shown in Table 7.3. The same combination also gives the best performance with the CODEC. This combination also shows an output of the CODEC that is a closer match to the original image than the filtered input. However, this is not always the case. For example, the Barbara image with 21dB of noise shows that the ASF filter structure with 8nn connectivity works best with the area, contrast and volume attributes. The noise reduction showed that the ASF filter structure worked best but in combination with 4nn connectivity. There appears to be no logical connection between the best filter for the noise reduction and which produces the best output from the CODEC. Thus the best performing noise reduction filter may not necessarily produce the optimum output from the CODEC.

Image (Noise level (dB))	Filtering method	PSNR (dB)
Barbara 14.48dB	Gaussian	**23.04**
	Morphological	22.19
Barbara 2 14.47dB	Gaussian	**23.01**
	Morphological	22.25
Boats 14.41dB	Gaussian	**24.99**
	Morphological	24.12
Goldhill 14.39dB	Gaussian	15.58
	Morphological	**24.63**
Simulated 14.47dB	Gaussian	29.62
	Morphological	**32.76**
Barbara 21.46dB	Gaussian	**26.29**
	Morphological	26.01
Barbara 2 21.49dB	Gaussian	**26.46**
	Morphological	26.36
Boats 21.43dB	Gaussian	**28.66**
	Morphological	28.29
Goldhill 21.47dB	Gaussian	**29.14**
	Morphological	28.15
Simulated 21.44dB	Gaussian	35.03
	Morphological	**38.08**
Barbara 28.09dB	Gaussian	30.24
	Morphological	**30.66**
Barbara 2 28.08dB	Gaussian	30.34
	Morphological	**30.90**
Boats 28.11dB	Gaussian	32.35
	Morphological	**32.76**
Goldhill 28.09dB	Gaussian	**32.35**
	Morphological	31.92
Simulated 28.01dB	Gaussian	37.24
	Morphological	**44.26**
Barbara 34.96dB	Gaussian	35.52
	Morphological	**36.19**
Barbara 2 34.97dB	Gaussian	35.60
	Morphological	**36.20**
Boats 34.99dB	Gaussian	36.52
	Morphological	**37.55**
Goldhill 34.96dB	Gaussian	**36.49**
	Morphological	36.37
Simulated 34.96dB	Gaussian	39.69
	Morphological	**51.33**

Table 7.4: Comparison of the best Gaussian and Morphological noise reduction results. Shaded cells show the best results for the given image.

In addition, in some cases the output of the CODEC results in a better PSNR compared with the noise reduction method using the same filter structure, connectivity and attribute. However this is not always the case. For example, using the volume attribute with 14dB of noise on the Barbara image, the ASF filter structure and 4nn connectivity results in a noise-reduced image with a PSNR of 21.81dB. Once compressed and decompressed the PSNR drops to 20.26dB. However using the power attribute with the ASF filter structure and 4nn connectivity, the noise-reduced image has a PSNR of 22.03dB whilst the output of the CODEC has a PSNR of 22.26dB. With the exception of the AF filter structure using 4nn connectivity and the contrast attribute, all of the outputs give a better PSNR than if the image had not been filtered at all. Thus this shows that filtering, regardless of the

combination of attribute, connectivity and structure will increase the compressibility for the JPEG CODEC. The results for the JPEG 2000 CODEC (see Table 7.6) show that the best filter for the noise reduction using the power attribute is for the majority of the time (87.5%) is also the best combination for the CODEC output, based on the Barbara, Barbara 2, Boats and Goldhill images. In filters used for the noise reduction using the area attribute show that the same combination will produce the best CODEC output 81.25% of the time. Thus showing that the filter that performs best is dependant upon the CODEC used. It is noted that the attribute values have been optimised to produce the optimum noise free image before compression.

7.1.3 Application to Unseen Data

Both the Gaussian and morphological filtering methods have been shown to remove noise very effectively. However, all of the test data used has the original uncorrupted image. In the real world, often only access to the noisy image is available. For example a television set only has access to the image it receives and not the original that is present in the studios. Hence it is required that a noisy image can be given to a preprocessing system and for it to filter the noise out using only statistics of the noisy image (the received image). There are several possible ways in which this could be accomplished. The chosen method makes use of the fact that a good estimate of the noise variance can be made with no prior knowledge (see section 5.2). Firstly the optimum attribute values are plotted against noise variance for the test data as shown for the Gaussian σ value, mask size in Figure 7.30 and Figure 7.31 respectively. Figure 7.32 shows the area size against noise variance for the 4nn connectivity and AF filter structure. The dots on the graphs show actual measured values and the solid lines show a quadratic fit to the data. The equation of the fit is obtained, with equations 7.1 and 7.2 show the quadratic equation for the Gaussian sigma and mask size respectively and the area size is given by equation 7.3. To calculate the attribute value to use, the noise variance is placed in the variable x. So for example, using a noise variance of 2312, which corresponds to the noise variance of the Barbara image with 14dB of noise present, then the appropriate area size to use in combination with the AF filter structure and 4nn connectivity, is found by replacing x in equation 7.3 with 2312. This gives an area size 41.42, which is rounded to the nearest whole number to give an area size of 41. The true optimum area size for this particular filter is 35 so using the estimated area size will slightly over filter the image in this case. However some images will be under-filtered whilst a few will be close to their optimum. The Gaussian σ value is rounded to one decimal place and the mask size is rounded to the nearest odd number and all morphological attributes are rounded to the nearest whole number. In addition, the absolute value of the equation is used to avoid any negative numbers.

$$y = -0.000000188x^2 + 0.0000918x + 0.459 \tag{7.1}$$

$$y = -0.0000356x^2 + 0.0134x + 2.49 \tag{7.2}$$

$$y = -0.0000047x^2 + 0.0286x + 2.06 \tag{7.3}$$

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara (Original noise free)			27.32						
Simulated (Original noise free)			45.02						
Barbara 14.48dB	Noisy		18.59						
	Area	35	21.94	56	**22.29**	438	21.79	441	21.79
	Contrast	80	18.77	181	21.39	80	18.83	237	21.18
	Volume	256	20.26	1123	20.26	256	20.17	8485	20.17
	Power	71210	22.15	70713	22.26	834790	21.83	839600	21.83
Simulated 14.47dB	Noisy		21.31						
	Area	157	27.56	808	**33.20**	1064	28.41	1064	28.41
	Contrast	199	24.00	180	25.31	199	24.00	180	24.78
	Volume	256	22.70	3761	22.70	256	22.70	8838	22.70
	Power	257623	27.61	71089	27.03	70978	25.99	71005	25.99
Barbara 21.46dB	Noisy		23.73						
	Area	8	24.76	8	24.77	18	24.79	18	24.79
	Contrast	45	24.62	58	24.88	45	24.70	59	24.93
	Volume	113	24.66	156	24.62	146	24.66	267	24.71
	Power	5780	**25.24**	5778	**25.24**	9550	25.22	9550	25.22
Simulated 21.44dB	Noisy		27.48						
	Area	84	34.87	1018	**37.72**	1046	34.88	1046	34.88
	Contrast	53	35.07	74	31.33	52	35.26	76	30.65
	Volume	255	29.60	1682	29.60	245	29.89	7220	29.89
	Power	654555	35.21	164823	35.41	479743	35.12	20870	33.31
Barbara 28.09dB	Noisy		26.45						
	Area	2	26.67	2	26.67	4	26.85	4	26.85
	Contrast	22	26.87	23	26.88	22	26.92	23	26.94
	Volume	37	26.87	39	26.86	44	**26.96**	55	26.92
	Power	793	26.91	793	26.90	973	**26.96**	973	**26.96**
Simulated 28.01dB	Noisy		33.71						
	Area	95	38.98	269	**41.60**	1060	38.51	1060	38.51
	Contrast	35	41.34	36	36.75	32	38.80	36	36.55
	Volume	254	36.20	1087	36.20	253	36.12	4749	36.12
	Power	14460	39.21	28985	40.16	52028	38.88	52028	38.88
Barbara 34.96dB	Noisy		27.20						
	Area	1	27.20	1	27.20	2	27.17	2	27.17
	Contrast	10	27.44	10	27.44	10	**27.45**	10	**27.45**
	Volume	12	27.19	12	27.19	13	27.19	14	27.18
	Power	116	**27.45**	116	**27.45**	136	**27.45**	136	**27.45**
Simulated 34.96dB	Noisy		39.14						
	Area	130	**44.52**	115	44.21	455	41.08	456	41.08
	Contrast	13	43.65	17	41.37	13	40.45	17	40.57
	Volume	234	41.55	510	41.55	253	40.37	2409	40.37
	Power	3066	**44.52**	3010	43.16	15553	41.08	15562	41.08

Table 7.5: Output PSNR of the optimum attribute values using JPEG at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst results.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara (Original noise free)			31.16						
Simulated (Original noise free)			55.33						
Barbara 14.48dB	Noisy		17.67						
	Area	35	21.92	56	22.20	438	21.46	441	21.44
	Contrast	80	18.13	181	21.62	80	18.20	237	20.98
	Volume	256	20.61	1123	20.61	256	20.14	8485	20.14
	Power	71209	22.19	70713	**22.22**	834790	21.47	839600	21.46
Simulated 14.47dB	Noisy		19.93						
	Area	157	27.33	808	**32.98**	1064	27.95	1064	27.95
	Contrast	199	19.01	180	25.98	199	19.01	180	24.66
	Volume	256	23.45	3761	23.45	256	22.59	8838	22.59
	Power	257623	27.82	71089	27.42	70978	25.84	71005	25.71
Barbara 21.46dB	Noisy		24.49						
	Area	8	25.41	8	25.41	18	25.19	18	25.19
	Contrast	45	24.83	58	25.39	45	24.99	59	25.60
	Volume	113	25.70	156	25.50	146	25.78	267	25.71
	Power	5780	**25.93**	5778	25.89	9550	25.79	9550	25.79
Simulated 21.44dB	Noisy		25.78						
	Area	84	35.54	1018	**38.21**	1046	34.83	1046	34.83
	Contrast	53	34.23	74	31.94	52	32.31	76	30.93
	Volume	255	30.20	1682	30.20	245	29.98	7220	29.98
	Power	654555	35.22	164823	35.57	479743	34.96	20870	33.92
Barbara 28.09dB	Noisy		28.87						
	Area	2	28.97	2	28.97	4	29.01	4	29.01
	Contrast	22	29.01	23	29.05	22	29.06	23	29.15
	Volume	37	29.07	39	29.12	44	29.15	55	29.12
	Power	793	29.18	793	29.18	973	**29.29**	973	**29.29**
Simulated 28.01dB	Noisy		33.14						
	Area	95	40.77	269	**43.97**	1060	38.84	1060	38.84
	Contrast	35	42.93	36	37.90	32	38.31	36	37.30
	Volume	254	36.71	1087	36.71	253	36.60	4749	36.60
	Power	14460	40.84	28985	41.65	52028	39.79	52028	39.79
Barbara 34.96dB	Noisy		30.49						
	Area	1	30.49	1	30.49	2	30.50	2	30.50
	Contrast	10	29.52	10	30.53	10	30.55	10	30.56
	Volume	12	30.56	12	30.56	13	30.51	14	**30.57**
	Power	116	30.55	116	30.56	136	30.55	136	30.54
Simulated 34.96dB	Noisy		39.57						
	Area	130	**51.32**	115	50.12	455	43.54	456	43.54
	Contrast	13	47.99	17	44.31	13	41.36	17	42.28
	Volume	234	44.22	510	44.22	253	41.50	2409	41.50
	Power	3066	**51.32**	3010	47.34	15553	43.54	15562	43.54

Table 7.6: Output PSNR of the optimum attribute values using JPEG 2000 at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst results.

Equations for the remaining filter attributes and combinations are given in Appendix B, section 13.1.4. Although the quadratic equation fits the data well, a much better fit, and therefore estimate of the attribute value, could be achieved by using much more data points. However, this would require a significantly larger number of images, 50 or more, and the optimum filters to be found for them over a range of attribute values with varying amounts of noise. Whilst this is relatively fast for the area and contrast attributes, the volume and power attributes take considerably longer due to the fact that they both can have extremely large attribute values, which increases the search range significantly. Thus, this research is only aimed at proving the use of morphological filtering for preprocessing and leaves refinement and optimisation of the implementation (source code) to future work.

7.1.3.1 Estimating Noise Variance

To estimate the attributes, the noise variance is used as an input. However, given a noisy image and no prior information, an estimate of the noise is made using the method described in section 5.2. Briefly, this breaks in image into blocks of size m x m and calculates the variance of each block [91]. A percentage of the blocks with the lowest variance are then used to calculate the noise variance by averaging them together. This however introduces two more variables that need to be chosen. Olsen suggest that the optimal block size is 7 x 7, but do not give any indication of the optimum percentage of blocks to use [91]. A simple series of experiments is carried out to determine the best values to use. This is done by using 53 images, and corrupting them with AWGN (see section 5.1.1) ranging from 42dB to 7dB. The block size is then iterated from 2 x 2 to 29 x 29, and the percentage from 1% to 100%. The true noise variance is known for this test data, so the best estimate can be easily found from the results. Figure 7.33 shows the number of times, as a percentage, that the best percentage to use and block size occur at. The results show that taking 2% of the smallest variances is the most frequently used (4.9% of the time) and that a 2 x 2 is the best block size to use, coming out the best 24% of the time.

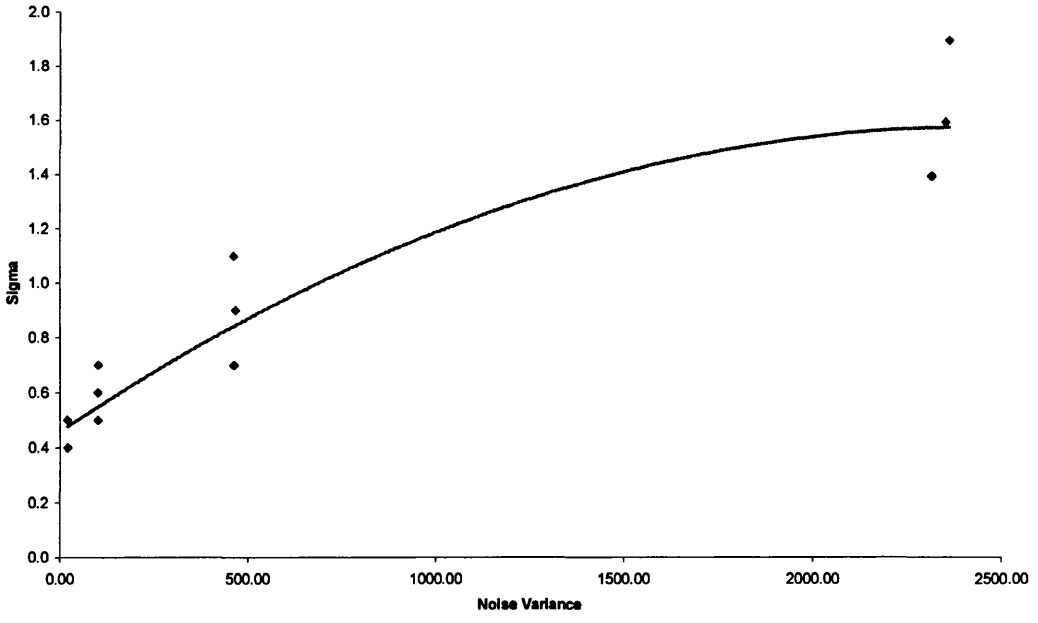


Figure 7.30: Gaussian sigma (σ) value against the noise variance. Points show the original measure data and the solid line shows a quadratic fit to the data.

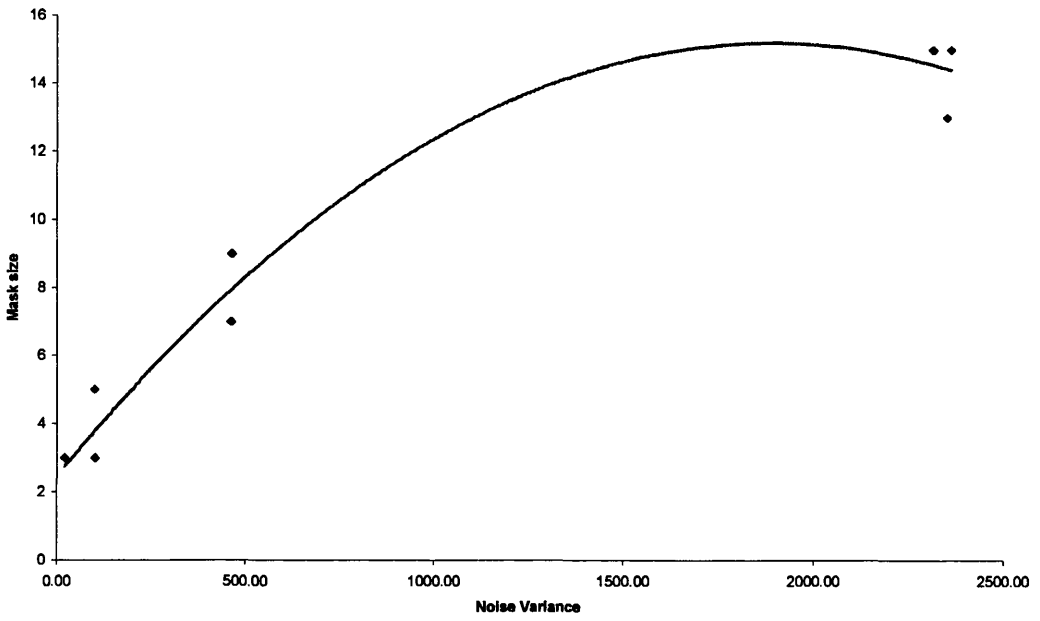


Figure 7.31: Gaussian mask value against the noise variance. Points show the original measure data and the solid line shows a quadratic fit.

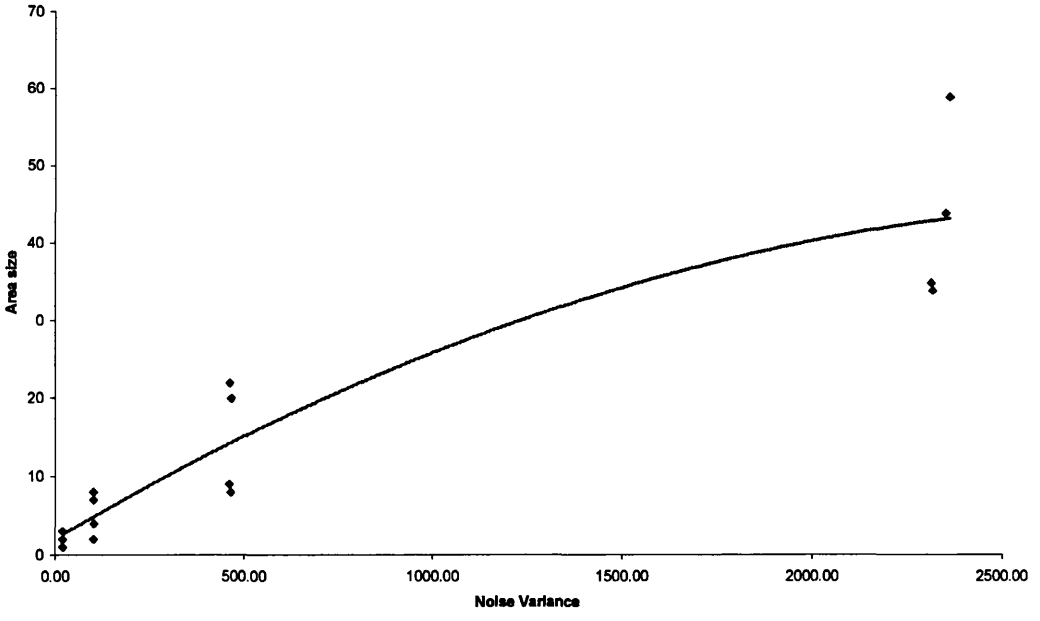
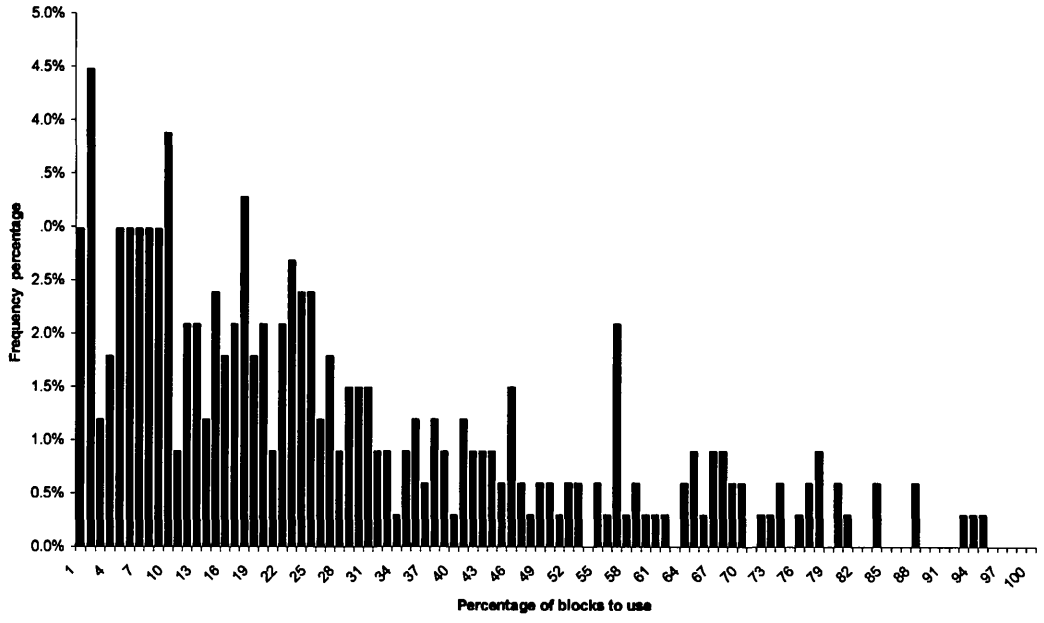
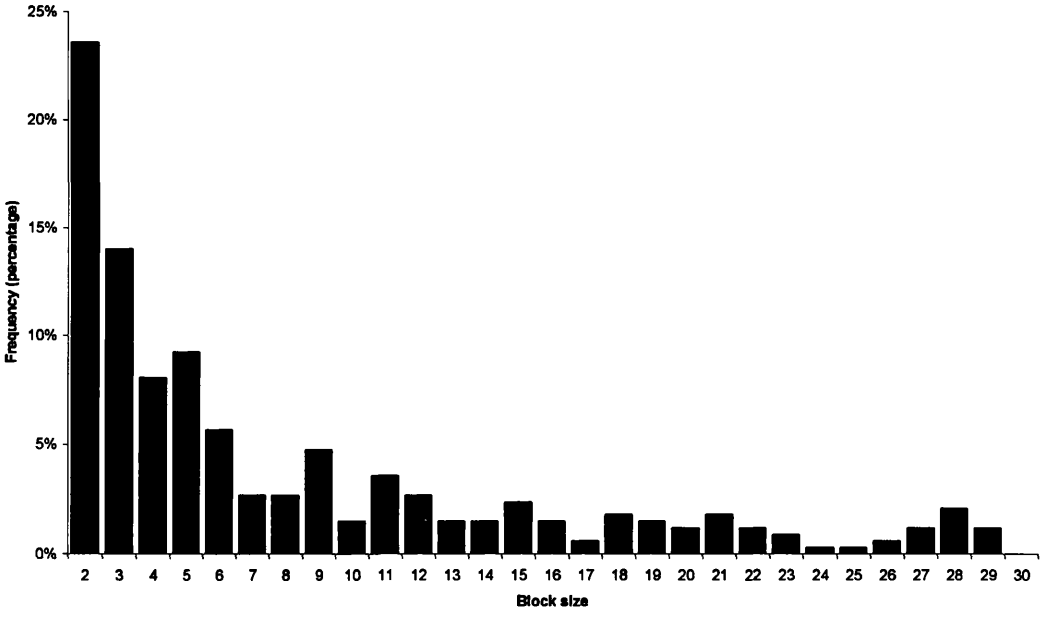


Figure 7.32: Area size value against the noise variance for 4nn connectivity and the AF filter structure. Points show the original measure data and the solid line shows a quadratic fit.



a) Percentage of blocks to take.



b) Frequency of the block size to use.

Figure 7.33: Frequency of the best percentage of results and block sizes to use.

However, the results that use a 2 x 2 may not necessarily be produced using 2% of the blocks, and similarly, the results using 2% of the blocks may not have been produced by a block size of 2 x 2. Thus two more experiments are carried out. The first uses a block size of 2 x 2, and finds the optimum percentage of blocks to use with it (see Figure 7.34) and the second uses takes a fixed percentage (2%) of the blocks and finds the optimum block size to use (see Figure 7.35). The results show that for a block size of 2 x 2, using 96% of the blocks with the lowest variance is the most frequently used (3.25% of the time) to obtain the best performance. However, using 2% of the blocks with the lowest variances, the results show that a 4 x 4 block size produces the optimum results. This gives two possible combinations, the first using a 2 x 2 block and taking the average of the lowest 96% of them whilst the second uses a 4 x 4 block and takes the average over only 2% of the blocks. Table 7.7 shows the average error between the true noise variance and the estimated using the above variables. The table shows the average errors for 6 different amounts of noise, showing that the higher the noise, the less accurate the estimate is. Using the first combination of 2 x 2 blocks and 96% of the blocks shows less error for the 3 high noise levels (7dB, 13dB and 20dB) where as the combination of 4 x 4 blocks and 2% of the blocks show less error for the low noise (27dB, 34dB and 42dB) images. Thus the first combination is used for noise estimation from this point forward as its error overall is lower than the second combination and also because this should result in a better estimate of the attribute value to use at high noise levels, where the filtering will be more noticeable.

Actual PSNR (dB) noise level	Average error between measured and true noise variance.	
	Using 2 x 2 block size and 96% of the blocks.	Using 4 x 4 block size and 2% of the blocks.
7	1867.81	3833.49
13	488.84	1460.73
20	153.58	307.88
27	102.01	52.91
34	95.01	11.71
42	94.37	9.5

Table 7.7: Average error in the noise estimation with various amounts of noise.

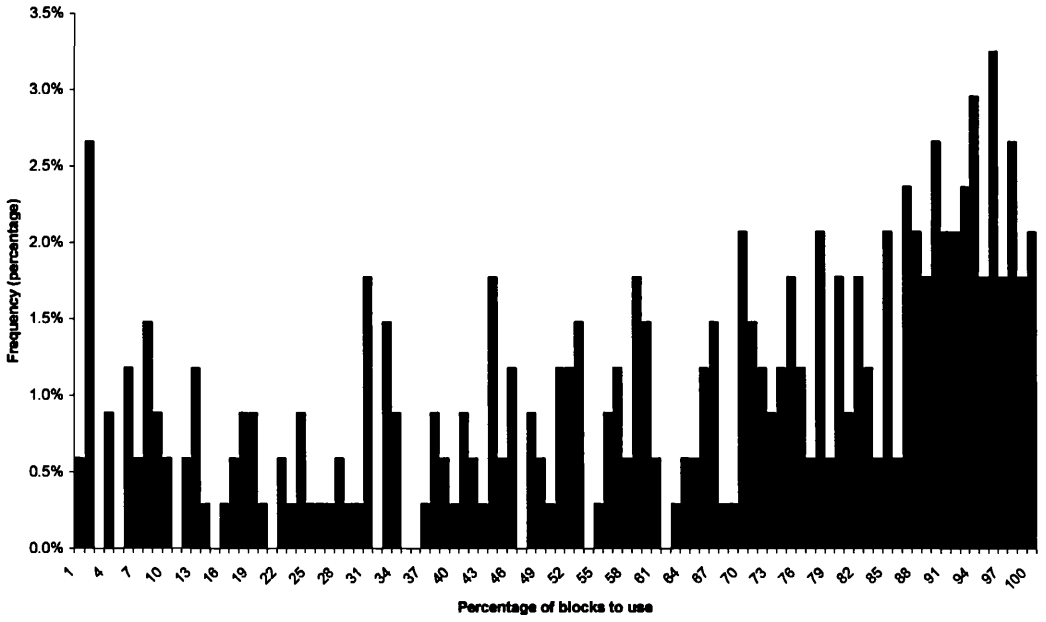


Figure 7.34: Frequency of percentage of blocks to use in conjunction with a 2 x 2 block size.

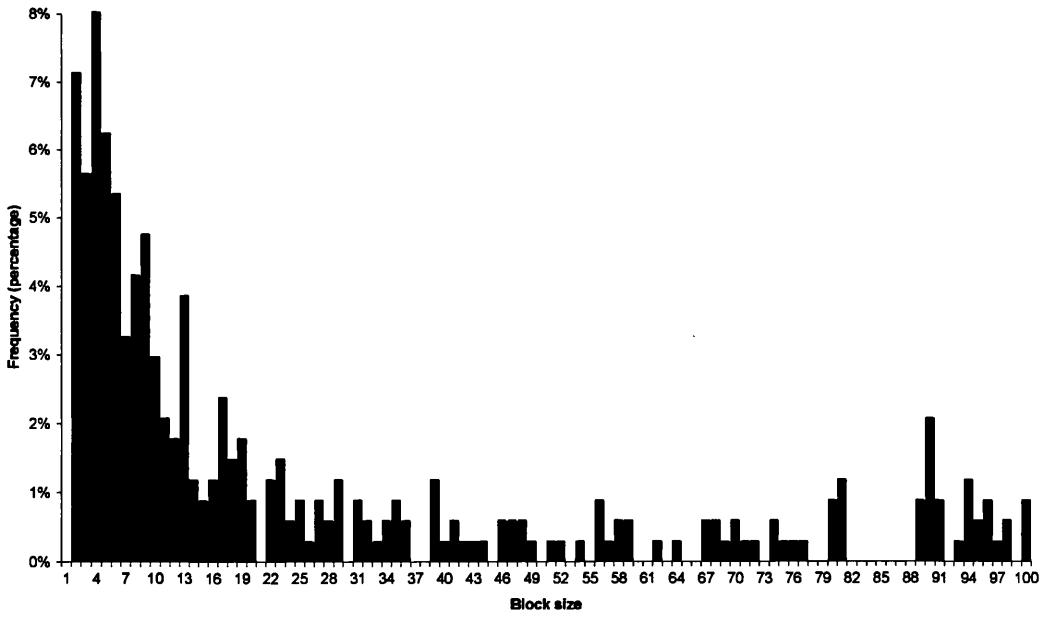


Figure 7.35: Frequency of block sizes to use in conjunction with using 2% of the blocks.

7.1.3.2 Applying Filters to Unseen Data

The attribute estimation equations (see section 7.1.3) can be used in conjunction with the noise estimation (see section 7.1.3.1) to form a completely '**blind**' system (i.e. no prior knowledge is required). True unknown data is data that has already been corrupted with noise, and is not used here, as the performance of the system cannot be measured, other than by visual inspection. Instead two images, Blackboard and Girl are used, which are corrupted with 4 different levels of AWGN (see section 5.1.1). The original images are then discarded for the filtering and only the noisy images are used. This method has the advantage that the original image can still be used to evaluate the performance of the system. An estimate of the noise variance in the noisy images is made (see Table 7.8), which shows the estimated values being very close to the actual noise variance. The estimated noise variance is then used, with the method and results found in start of section 7.1.3, to form an estimate of the attribute values for each filter, which are shown in Tables 7.9 and 7.10 for the Blackboard and Girl images using the Gaussian and morphological filters respectively. Filtering is then applied to the noisy images using the estimated attribute values, after which the PSNR value is measured using the original uncorrupted image (see Tables 7.9 and 7.10). All of the filtering results in an improvement in the PSNR. The Gaussian results give a better performance than the morphological for the Girl image with all amounts of noise and for the Blackboard image with 14dB of noise. However, both sets of results are very close, for example the Blackboard image with 14dB of noise differs only by 0.27dB.

Image	True values		Estimated noise variance
	PSNR (dB)	Noise variance	
Blackboard	14.07	2528.50	2265.00
	21.06	508.16	496.83
	27.98	102.82	118.71
	34.92	20.26	38.11
Girl	14.08	2537.10	2277.11
	20.91	526.9	500.43
	27.99	102.53	117.29
	34.91	20.25	40.22

Table 7.8: Comparison between estimated and true noise variance measurements for the 8bit greyscale, 720 x 576 Blackboard and Girl images.

The morphological results, with the exception of the Girl image with 35dB of noise, all show that the best filtered output is produced using the power attribute and 4nn connectivity. In addition, the results also support the fact that the ASF filter structure works better with a high amount of noise (14dB) and that the AF filter structure works better for lower amounts of noise (21dB, 28dB and 35dB). This has shown that using an image corrupted with noisy and with no prior knowledge, an image can be successfully filtered to reduce the noise present using morphological filtering. In addition this has shown that a good estimate of the attributes can be obtained, and although they may not be the best

values to use, will provide a good increase in PSNR. Figures 7.36 and 7.37 show the results of filtering the Blackboard image with 14dB and 35dB of noise respectively with the Gaussian filter. The images show that although a large increase in PSNR is achieved, especially with the images with a high amount of noise (14dB), that the images are blurred. This is less noticeable on low noise images (35dB) as a smaller mask size reduced the effects of the Gaussian filtering. It is arguable as to, which is the most visually acceptable, the original noisy image or the blurred filtered version. In addition, the brightness of the final output appears to be brighter than on the input, which is noticeable visually, but only when compared to the original. Figure 7.38 shows the Blackboard image and two versions corrupted with 14dB and 35dB of AWGN (see section 5.1.1). The filtered versions are shown in Figures 7.39 and 7.40 for 14dB and 35dB of noise respectively. The images corrupted with 14dB of AWGN and filtered (see Figure 7.39) show a significant reduction in noise, both numerically and visually. Visually, the Gaussian (see Figure 7.36) appears to distort more of the detail than the morphological, although psychovisual evaluations would need to be carried out to determine if this is the case. The worst filters, for the Blackboard image (see Figure 7.39c) removes a significant amount of detail from the image. Using the images corrupted with 35dB of AWGN, it is much harder to see a difference between the filtered images (see Figure 7.40).

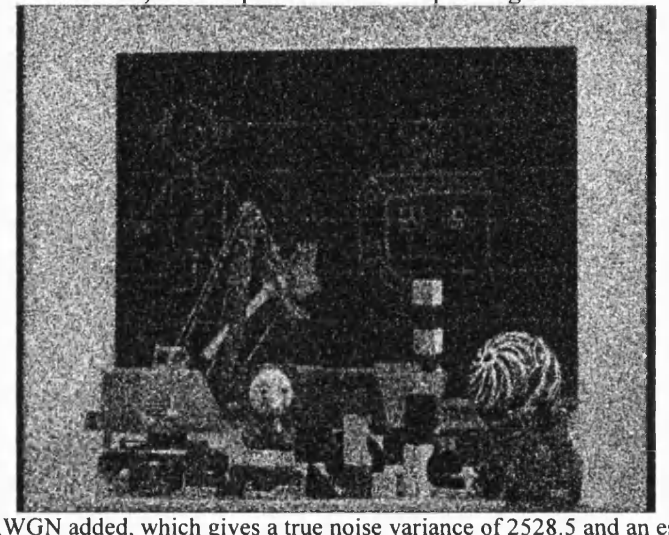
Image (noise level (dB))	Estimated σ	Estimated mask size	PSNR (dB)
Blackboard (14.07dB)	2.5	13	23.88
Girl (14.08dB)	2.6	13	25.80
Blackboard (21.06dB)	0.9	9	28.85
Girl (20.91dB)	0.9	9	29.52
Blackboard (27.98dB)	0.6	5	32.45
Girl (27.99dB)	0.6	5	33.00
Blackboard (34.92dB)	0.5	3	36.81
Girl (34.91dB)	0.5	3	37.40

Table 7.9: Estimated Gaussian filter variable values for the 8bit greyscale, 720 x 576 Blackboard and Girl images.

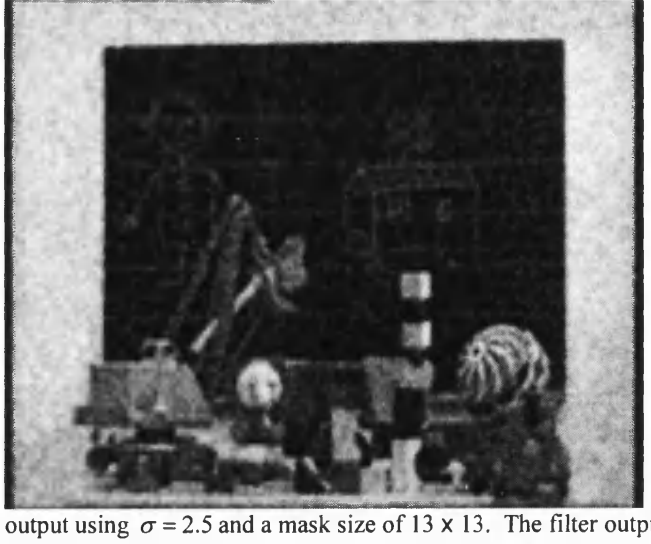
Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Blackboard (14.07dB)	Area	41	20.01	15	24.18	5	21.25	3	21.24
	Contrast	84	15.43	46	21.76	21	15.99	15	19.42
	Volume	268	18.94	173	23.07	54	17.03	24	20.80
	Power	46085	23.35	9050	**23.61**	1740	21.46	501	21.49
Girl (14.08dB)	Area	41	20.59	15	24.88	5	21.80	3	21.79
	Contrast	83	17.62	46	21.65	21	17.57	15	19.23
	Volume	266	19.16	174	23.63	54	17.24	25	21.21
	Power	76731	24.00	9132	**24.22**	1717	21.87	532	21.89
Blackboard (21.06dB)	Area	81	25.86	15	28.47	5	26.64	4	26.77
	Contrast	197	25.10	70	27.60	26	24.74	15	25.59
	Volume	1850	25.98	128	28.33	72	24.19	51	26.82
	Power	124066	**28.91**	7016	28.51	1793	26.83	1582	25.32
Girl (20.91dB)	Area	82	25.64	15	28.22	5	26.35	4	26.49
	Contrast	198	25.37	70	27.34	25	24.57	16	25.27
	Volume	1867	25.92	220	28.06	72	24.09	51	26.44
	Power	125394	**28.48**	7100	28.15	1786	26.38	1583	24.78
Blackboard (27.98dB)	Area	769	30.90	40	32.51	10	31.67	9	31.59
	Contrast	77	32.61	45	32.87	21	31.83	15	31.55
	Volume	277	32.24	222	32.36	67	30.88	28	32.23
	Power	1026168	**33.95**	15982	33.79	3202	32.56	18320	31.90
Girl (27.99dB)	Area	777	30.54	41	32.07	10	31.09	9	31.03
	Contrast	77	31.95	46	32.18	21	31.12	15	31.01
	Volume	273	31.85	223	32.60	67	30.52	29	31.43
	Power	1039740	**32.90**	15755	32.79	3435	31.60	17874	31.30
Blackboard (34.92dB)	Area	483	36.51	62	37.54	9	37.06	1	34.92
	Contrast	226	37.96	68	38.02	25	37.50	15	37.36
	Volume	9281	37.84	701	38.07	123	37.06	58	37.55
	Power	1213886	**38.38**	3403	37.65	921	35.07	13900	35.50
Girl (34.91dB)	Area	487	36.16	63	37.03	9	36.31	1	34.91
	Contrast	337	36.60	68	37.03	25	36.35	15	36.56
	Volume	9373	**37.15**	708	37.07	122	36.48	59	36.61
	Power	1229259	37.14	2997	36.20	1113	33.08	13505	33.51

Table 7.10: Estimated morphological filter variable values and the resultant PSNR for the 8bit greyscale Blackboard and Girl images. Shaded cells show the best output for that particular image.

a) Uncorrupted Blackboard input image.



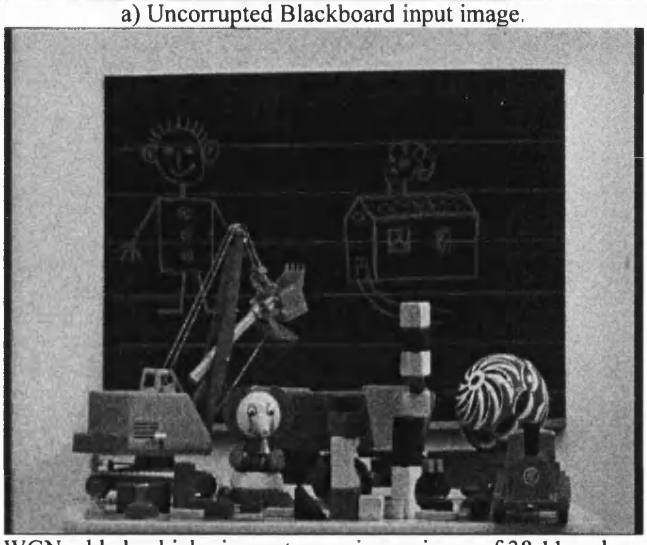
b) 14.07dB of AWGN added, which gives a true noise variance of 2528.5 and an estimated 2265.0.



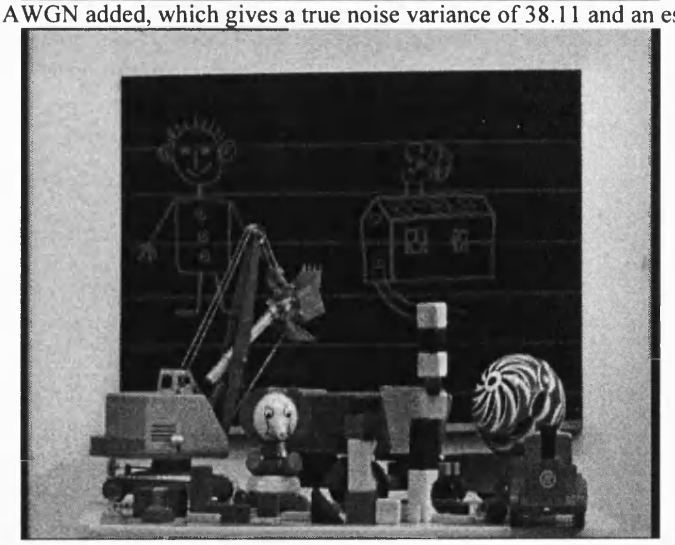
c) Gaussian filter output using $\sigma = 2.5$ and a mask size of 13 x 13. The filter output has a PSNR of 23.88 dB, which is an improvement of 9.8dB.

Figure 7.36: Corrupted and filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) using 14dB of noise.

a) Uncorrupted Blackboard input image.



b) 34.92dB of AWGN added, which gives a true noise variance of 38.11 and an estimated 20.26.

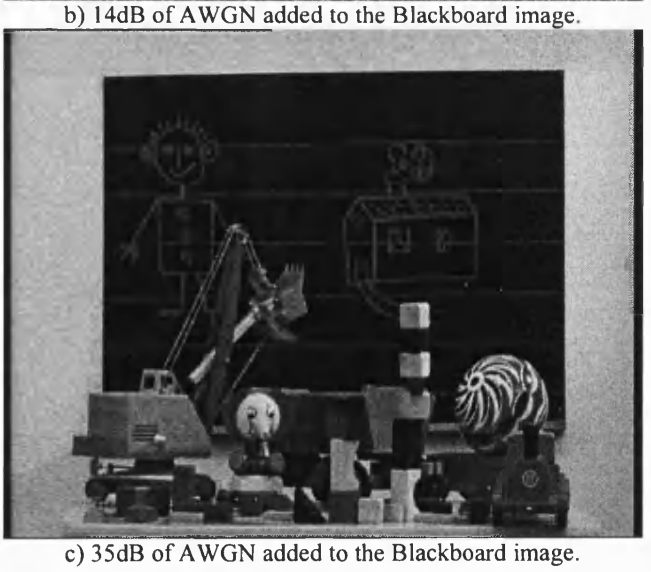


c) Gaussian filter output using $\sigma = 0.5$ and a mask size of 3 x 3. The filter output has a PSNR of 36.81 dB, which is an improvement of 1.89dB.

Figure 7.37: Corrupted and filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) using 35dB of noise.

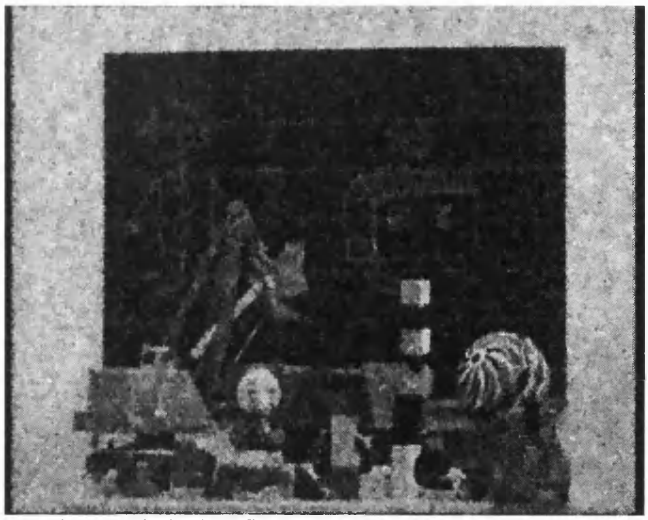
a) Original uncorrupted Blackboard image.

b) 14dB of AWGN added to the Blackboard image.

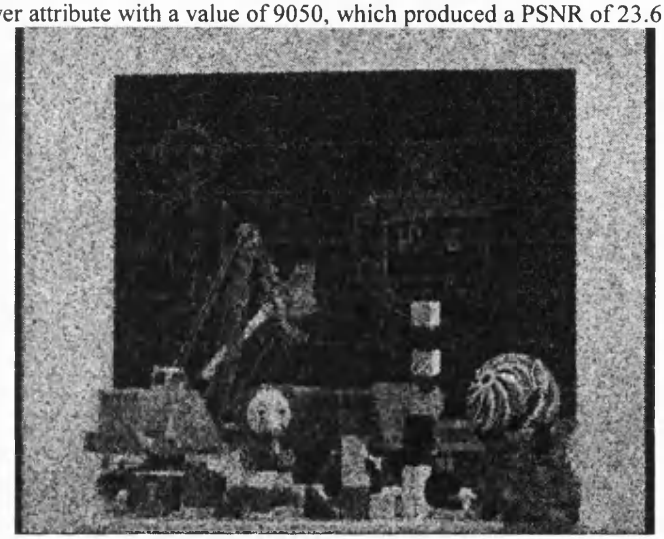


c) 35dB of AWGN added to the Blackboard image.

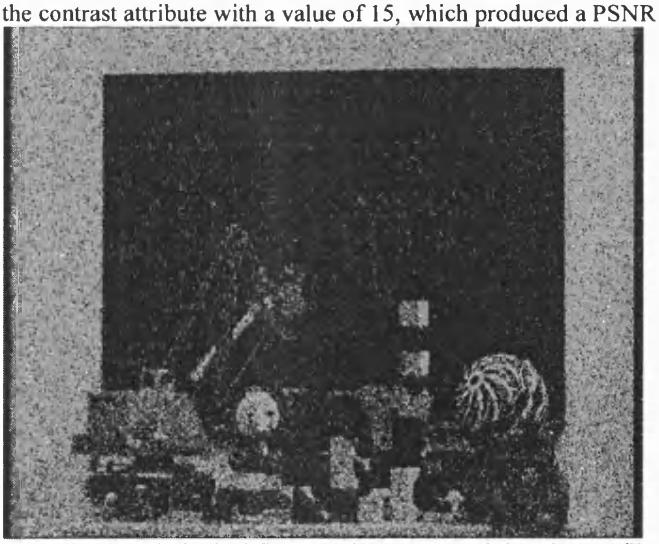
Figure 7.38: Corrupted input images of Blackboard (720 x 576, 8bit greyscale) with AWGN of 14dB and 35dB that are used as the input to the morphological filter.



a) Best filtered output. The morphological filter used 4nn connectivity, the ASF filter structure and the power attribute with a value of 9050, which produced a PSNR of 23.61dB.

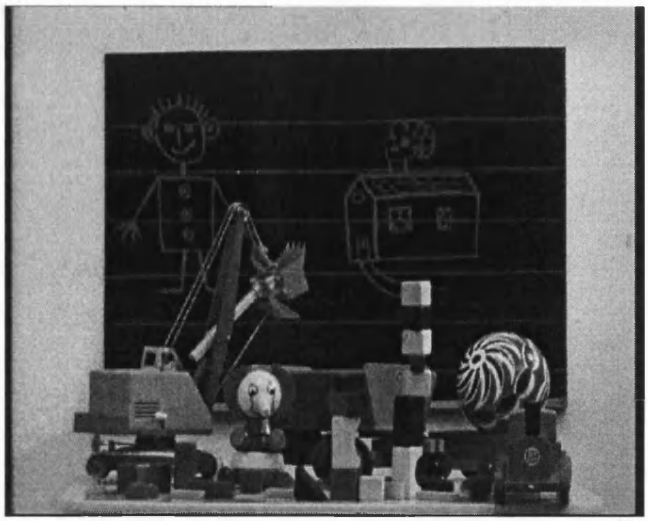


b) Filtered output lying between the best and worst outputs using 8nn connectivity, the ASF filter structure and the contrast attribute with a value of 15, which produced a PSNR of 19.42dB.

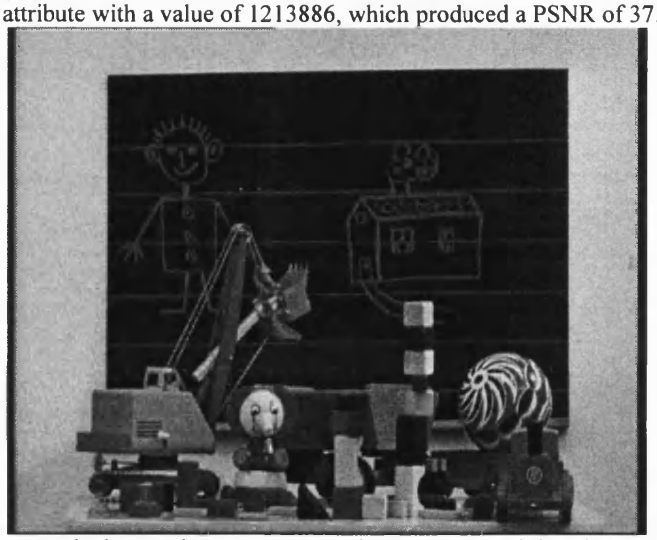


c) Worst filtered output. The morphological filter used 4nn connectivity, the AF filter structure and the contrast attribute with a value of 84, which produced a PSNR of 15.43dB.

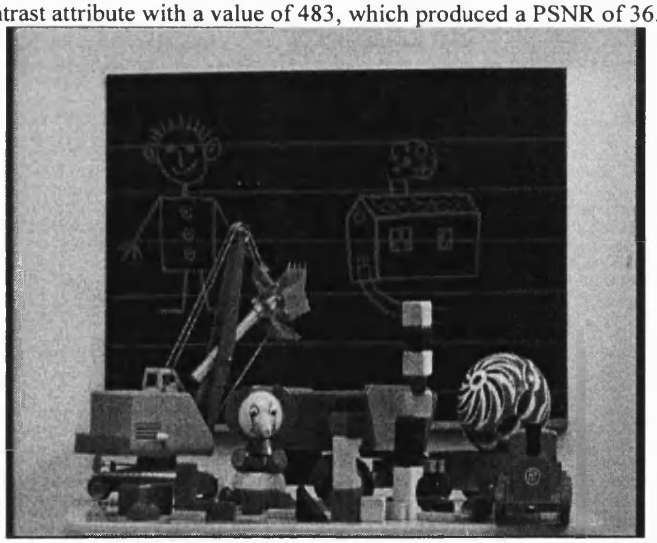
Figure 7.39: Filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) corrupted with 14dB of noise and using morphological filtering.



a) Best filtered output. The morphological filter used 4nn connectivity, the AF filter structure and the power attribute with a value of 1213886, which produced a PSNR of 37.15dB.



b) Result lying between the best and worst outputs using 4nn connectivity, the AF filter structure and the contrast attribute with a value of 483, which produced a PSNR of 36.51dB.



c) Worst filtered output. The morphological filter used 8nn connectivity, the ASF filter structure and the area attribute with a value of 1, which produced a PSNR of 34.92dB.

Figure 7.40: Filtered versions of the Blackboard image (8bit greyscale at 720 x 576 pixels) corrupted with 35dB of noise and using morphological filtering.

7.1.3.3 Compression Results

As the results for filtering may not be the optimum solution, the filtered results are compressed using both JPEG and JPEG 2000 CODEC's to ensure that they still improve the compressibility of the images as they did using the optimum filtering solutions (see section 7.1.2.1). The Gaussian results for both JPEG and JPEG 2000 are given in Table 7.11. The noise having been reduced should mean that the quality (PSNR) of the compressed images would approach that of the uncorrupted image. So for example, the original Blackboard image has a PSNR of 35.98dB when compressed to a ratio of 20:1 (0.4bpp). When the 14dB-corrupted version is compressed, it has a resultant PSNR of only 18.25dB using the JPEG CODEC. This shows that the CODEC removes some of the noise, yet the filtered version produces a PSNR of 21.22dB after compression. A gain is only achieved for the two highest noise levels (14dB and 21dB), showing that the Gaussian works better with high amounts of noise.

Image and noise level (dB)	PSNR (dB)			
	Un-filtered images		Filtered images	
	JPEG	JPEG 2000	JPEG	JPEG 2000
Blackboard (Original noise free)	35.98	38.68		
Girl (Original noise free)	34.01	35.91		
Blackboard 14.07	18.25	17.60	21.22	21.25
Girl 14.08	18.76	17.49	19.73	19.73
Blackboard 21.06	25.24	24.63	25.62	25.81
Girl 20.91	25.28	24.23	25.83	26.06
Blackboard 27.98	31.19	31.33	28.86	29.10
Girl 27.99	30.85	31.03	23.14	23.19
Blackboard 34.91	34.66	36.00	29.45	29.61
Girl 34.91	33.24	34.46	22.66	22.60

Table 7.11: JPEG and JPEG 2000 compression of the Gaussian filtered Blackboard and Girl image (8 bit greyscale at 720 x 576 pixels) using a compression ratio of 20:1 (0.4bpp).

Tables 7.12 and 7.13 show the results for morphological filtering followed by compression for the JPEG and JPEG 2000 CODEC's respectively. A compression ratio of 20:1 (0.4bpp) was used. The best combinations always performed better than had no filtering been done. For example using the Blackboard image with 14dB of noise present, the best filter produced a PSNR of 24.67dB. Using no filtering a PSNR of 18.25dB is achieved, thus filtering for this case has given a 6.42dB increase in performance. As shown previously (see section 7.1.2), the area attribute with the ASF filter structure and 4nn connectivity provides the best output performance for the highest amount of noise (14dB) and outperforms the Gaussian. The Gaussian output for the Blackboard image at 14dB was 21.22dB, thus the morphological output has a 3.45dB gain over it.

All of the morphological results above 14dB show that the best filter is produced by using the 4nn connectivity, the AF filter structure and the power attribute with no exceptions. In addition all of the best morphological filters perform better than their respective Gaussian filter. In most cases, the Gaussian filter is closer or worse than the worst filter obtained using the morphological methods. For example using the JPEG CODEC and 35dB of noise, the worst results for the morphological filtering are 32.39dB and 31.84dB for the Blackboard and Girl image respectively. Results for the JPEG CODEC using Gaussian filtering for the Blackboard and Girl images at a noise level of 35dB were 29.45dB and 22.66dB respectively. This shows that the morphological system is better at simplifying and removing noise to increase compressibility than the Gaussian.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Blackboard			35.98						
Girl			34.01						
Blackboard (14.07dB)	Noisy		18.25						
	Area	41	22.98	15	**24.67**	5	23.17	3	23.38
	Contrast	84	16.16	46	22.80	21	16.71	15	22.28
	Volume	268	20.75	173	23.97	54	20.54	24	23.36
	Power	46085	24.05	9050	24.44	1740	23.52	501	23.51
Girl (14.08dB)	Noisy		18.76						
	Area	41	23.64	15	**25.38**	5	23.90	3	24.03
	Contrast	83	18.92	46	22.88	21	19.30	15	22.33
	Volume	266	21.00	174	24.54	54	20.75	25	23.94
	Power	76731	24.65	9132	25.04	1717	24.08	532	24.05
Blackboard (21.06dB)	Noisy		25.24						
	Area	81	28.47	15	29.13	5	28.84	4	28.74
	Contrast	197	25.50	70	28.55	26	25.43	15	28.19
	Volume	1850	27.32	128	29.12	72	27.18	51	29.05
	Power	124066	**29.54**	7016	29.32	1793	29.17	1582	28.00
Girl (20.91dB)	Noisy		25.28						
	Area	82	28.31	15	28.95	5	28.56	4	28.48
	Contrast	198	26.48	70	28.38	25	26.36	16	27.99
	Volume	1867	27.31	220	28.90	72	27.19	51	28.65
	Power	125394	**29.17**	7100	28.99	1786	28.72	1583	27.91
Blackboard (27.98dB)	Noisy		31.19						
	Area	769	32.43	40	32.58	10	32.91	9	32.85
	Contrast	77	33.04	45	32.95	21	33.05	15	32.91
	Volume	277	32.42	222	33.12	67	32.46	28	33.24
	Power	1026168	**33.56**	15982	33.46	3202	33.40	18320	33.20
Girl (27.99dB)	Noisy		30.85						
	Area	777	31.83	41	31.89	10	32.05	9	32.04
	Contrast	77	32.08	46	32.16	21	32.08	15	32.07
	Volume	273	31.87	223	32.20	67	31.84	29	32.16
	Power	1039740	**32.39**	15755	32.35	3435	32.27	17874	32.16
Blackboard (34.92dB)	Noisy		34.66						
	Area	483	34.97	62	35.04	9	35.21	1	34.66
	Contrast	226	35.40	68	35.36	25	32.39	15	35.35
	Volume	9281	35.18	701	35.35	123	35.18	58	35.39
	Power	1213886	**35.47**	3403	35.28	921	33.89	13900	34.21
Girl (34.91dB)	Noisy		33.24						
	Area	487	33.45	63	33.55	9	33.59	1	33.24
	Contrast	337	33.54	68	33.64	25	33.62	15	33.69
	Volume	9373	33.58	708	33.59	122	33.59	59	33.64
	Power	1229259	**33.73**	2997	33.47	1113	31.84	13505	32.13

Table 7.12: Output PSNR of the morphological filters using unknown data with JPEG at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst reults.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Blackboard			36.68						
Girl			35.91						
Blackboard (14.07dB)	Noisy		17.60						
	Area	41	22.33	15	**24.49**	5	22.74	3	22.90
	Contrast	84	15.71	46	23.05	21	16.30	15	21.56
	Volume	268	21.16	173	23.92	54	20.58	24	22.63
	Power	46085	24.06	9050	24.24	1740	23.06	501	23.06
Girl (14.08dB)	Noisy		17.49						
	Area	41	22.93	15	**25.31**	5	23.49	3	23.63
	Contrast	83	18.23	46	23.42	21	18.68	15	21.80
	Volume	266	21.40	174	24.53	54	20.93	25	23.29
	Power	76731	24.72	9132	24.87	1717	23.73	532	23.67
Blackboard (21.06dB)	Noisy		24.63						
	Area	81	28.12	15	29.33	5	28.34	4	28.25
	Contrast	197	25.37	70	28.79	26	25.25	15	28.03
	Volume	1850	27.86	128	29.29	72	27.36	51	28.54
	Power	124066	**29.65**	7016	29.35	1793	28.64	1582	27.89
Girl (20.91dB)	Noisy		24.23						
	Area	82	27.88	15	29.15	5	27.88	4	27.73
	Contrast	198	26.03	70	28.54	25	26.06	16	27.72
	Volume	1867	27.82	220	29.14	72	27.35	51	27.98
	Power	125394	**29.33**	7100	29.18	1786	28.01	1583	27.80
Blackboard (27.98dB)	Noisy		31.33						
	Area	769	33.21	40	33.46	10	33.31	9	33.31
	Contrast	77	33.25	45	33.51	21	33.19	15	33.44
	Volume	277	33.45	222	33.78	67	33.23	28	33.50
	Power	1026168	**34.02**	15982	33.86	3202	33.45	18320	33.58
Girl (27.99dB)	Noisy		31.03						
	Area	777	32.58	41	32.72	10	32.51	9	32.54
	Contrast	77	32.41	46	32.65	21	32.38	15	32.62
	Volume	273	32.70	223	32.66	67	32.56	29	32.50
	Power	1039740	**32.77**	15755	32.63	3435	32.46	17874	32.67
Blackboard (34.92dB)	Noisy		36.00						
	Area	483	36.48	62	36.55	9	36.41	1	36.00
	Contrast	226	36.81	68	36.70	25	32.78	15	36.63
	Volume	9281	36.73	701	36.66	123	36.63	58	36.58
	Power	1213886	**36.86**	3403	36.48	921	34.50	13900	34.84
Girl (34.91dB)	Noisy		34.46						
	Area	487	34.64	63	34.67	9	34.46	1	34.46
	Contrast	337	34.42	68	**34.69**	25	34.37	15	34.63
	Volume	9373	34.72	708	34.61	122	34.64	59	34.58
	Power	1229259	34.67	2997	34.24	1113	32.04	13505	32.40

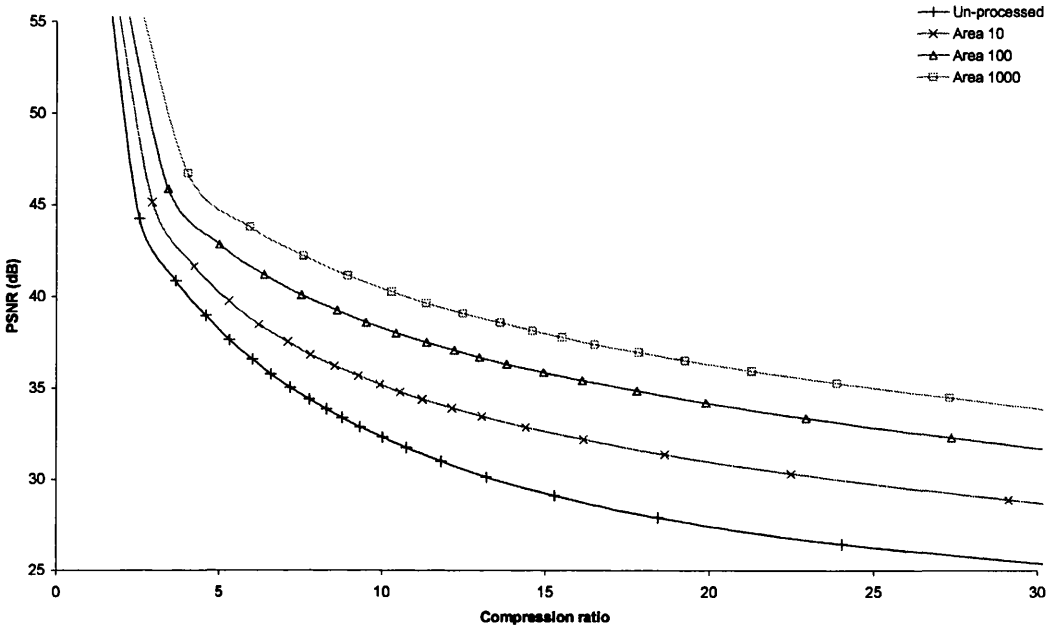
Table 7.13: Output PSNR of the morphological filters using unknown data with JPEG 2000 at a compression ratio of 20:1 (0.4bpp). The shaded values show the best output and the outlined cells show the worst reults

7.2 Psychovisually Lossless Simplification

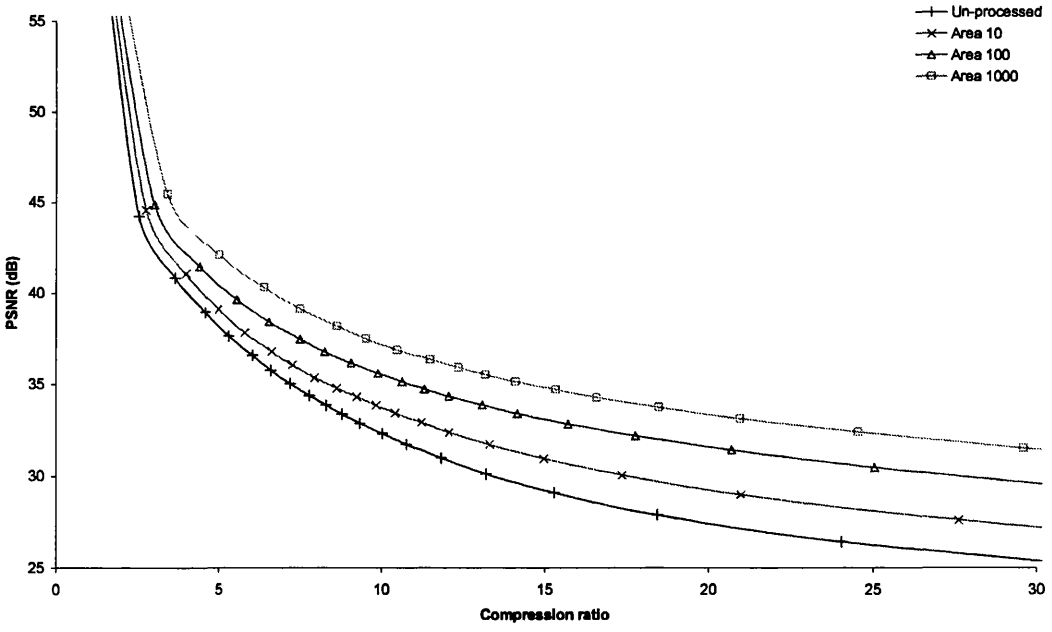
The previous section (section 7.1) showed how the performance of the JPEG and JPEG 2000 CODEC's could be improved by reducing the noise present within an image. The objective of this section is to produce a preprocessor that will simplify the image by removing only psychovisually redundant information and thus reducing the amount of information for the CODEC to compress. However this must also ensure that there is no visible degradation to the image. Section 6.2.2.4 proposed that the power attribute was a closer match to the HVS than any other attribute. Section 7.1 showed that the power attribute was also the best for noise removal and hence, only the standard area and power attributes are considered in this section. The filter structure and connectivity are investigated further with the use of psychovisual evaluations. A preprocessing system is developed in three stages; first the images are filtered and compressed for a range of attribute values and filter combinations to ensure that the resultant images are more compressible. Secondly the output of the preprocessor is tuned to be psychovisually lossless and then the output of the CODEC is evaluated to determine the amount of compression improvement whilst remaining visually lossless after compression.

7.2.1 Simplification

Since attribute morphology has not been widely adopted for image preprocessing, its effectiveness is evaluated for image simplification. Four ITU test images (Barbara, Barbara 2, Boats and Goldhill) are filtered using both the AF and ASF filter structures, 4nn and 8nn connectivity and the area attribute. The resultant images are then compressed and decompressed using JPEG and JPEG 2000 to produce rate distortion graphs. The results for the Barbara image are shown in Figure 7.41 and Figure 7.43 for JPEG and JPEG 2000 respectively. Results for the remaining three images are shown in Appendix B, Figures 13.37 – 12.48 (see section 13.2). As can be seen, any amount of filtering produces an improvement in the compression ratio for a given PSNR. The more filtering, the greater the improvement. This has proven that AM can be used for improving the compressibility of images using simplification.

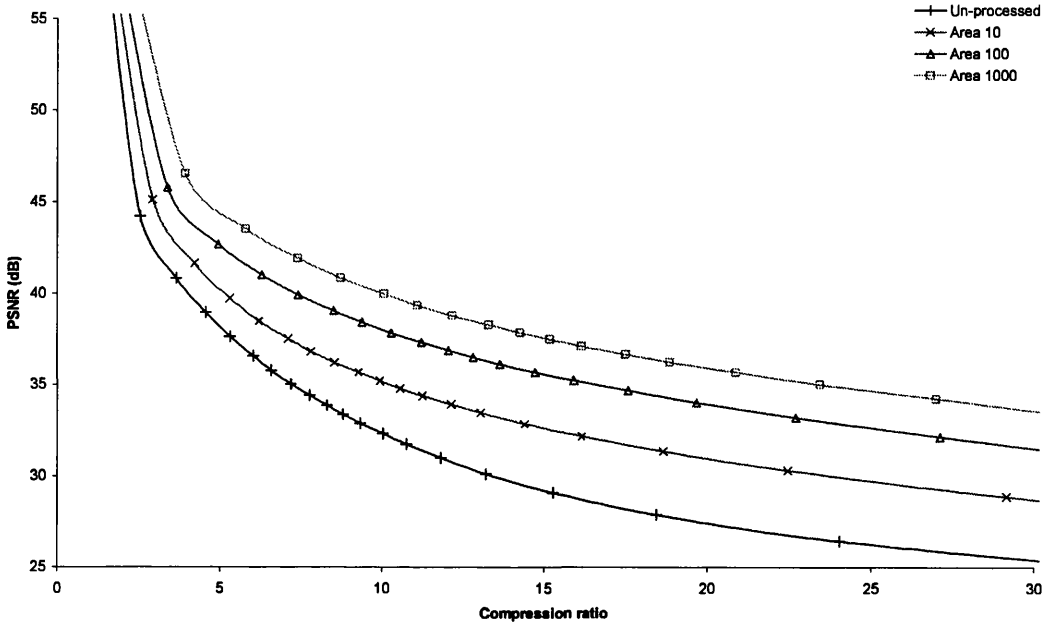


a) 4nn AF

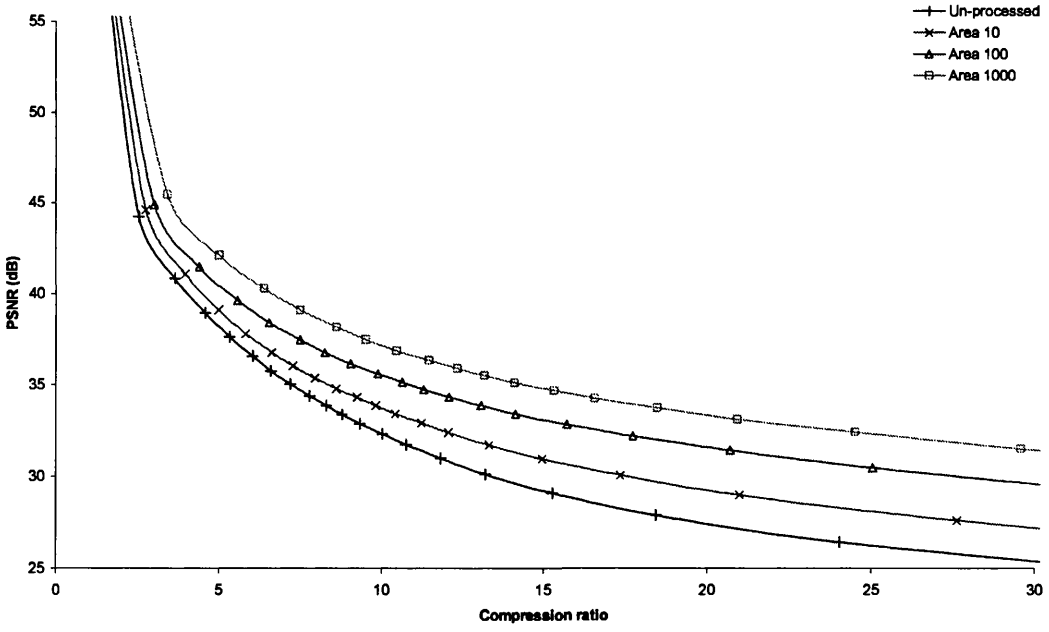


b) 8nn AF

Figure 7.41: JPEG rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the AF filter structure.

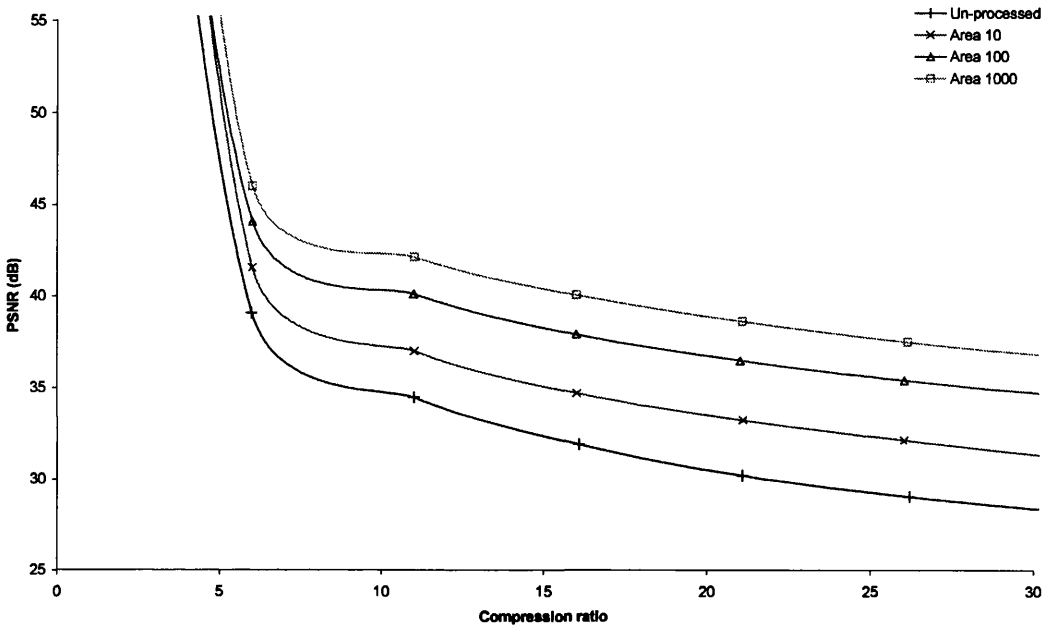


a) 4nn ASF

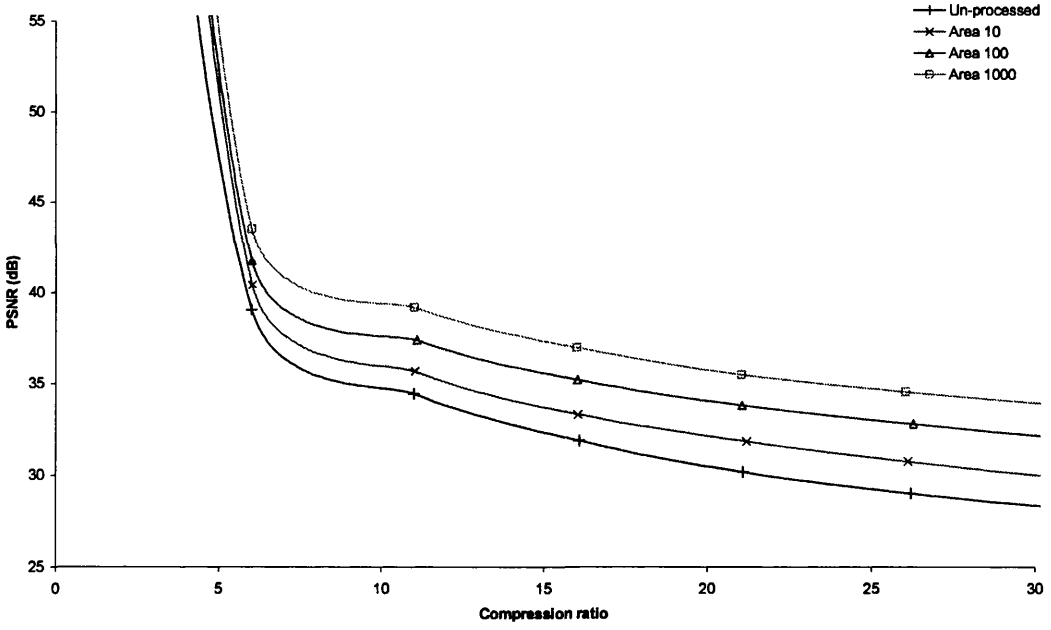


b) 8nn ASF

Figure 7.42: JPEG rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the ASF filter structure.

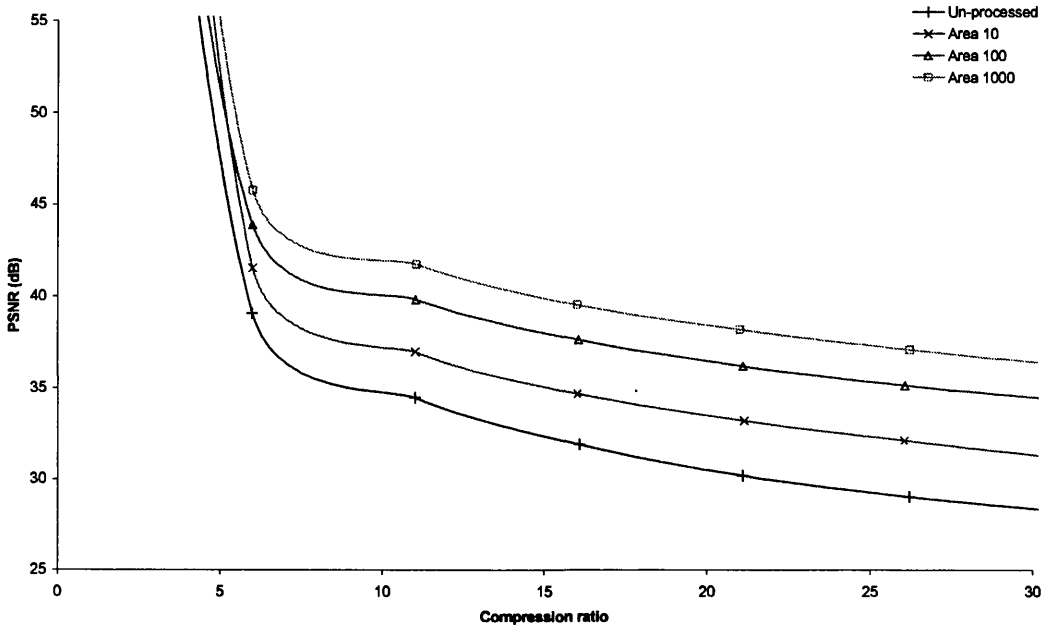


a) 4nn AF

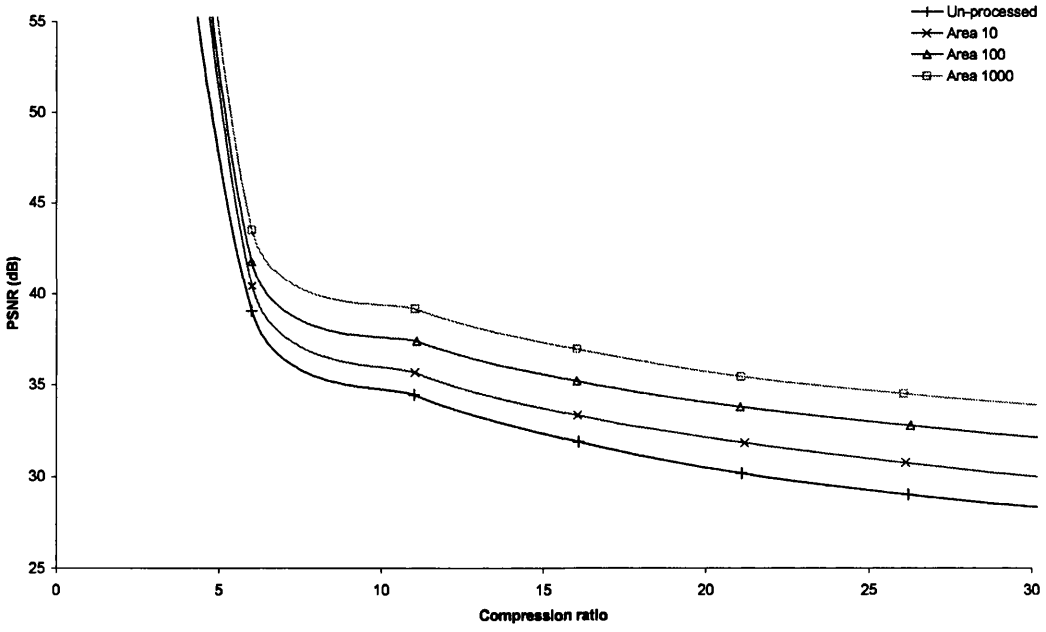


b) 8nn AF

Figure 7.43: JPEG 2000 rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the AF filter structure.



a) 4nn ASF



b) 8nn ASF

Figure 7.44: JPEG 2000 rate distortion graphs for the 8bit greyscale, 720 x 576 Barbara image using the ASF filter structure.

7.2.2 Psychovisual Determination of Visually Lossless Attribute Values

Chapter 6 showed that AM (see section 4.6.1) can be used to filter images to a high degree and still retain detail in the image, as shown for the Barbara image in Figure 6.3. Although this image still has detail left in it, most of the important information has been lost. Thus the image needs to be filtered as much as possible without deteriorating the visual quality. This is accomplished by finding an attribute value that results in a filtered image where filtering cannot be seen. One way to find the visually lossless attribute, lossless meaning that an observer cannot tell the difference, is to take an objective measure such as the PSNR between the input to the preprocessor and the output. However, the visually lossless value for the PSNR is dependant on the image and hence only propagates the problem to finding a value for each PSNR measurement that can be called visually lossless.

Thus psychovisual evaluations (see section 3.2) are used to find the optimum value for each attribute and filter structure. The base method for the psychovisual testing used is the DSIS method. Essentially the original and the impaired (preprocessed) images are individually displayed alternatively twice, the order being chosen randomly, and the observer asked to vote on the quality of one compared to the other. However, in some cases the observer may think that the impaired image is better than the original and this has to be accommodated. To achieve this, the double stimulus method is paired with the comparison scale to give the Double Stimulus, Comparison Scale (DSCS) method. A range of values for both the area and power attributes (see Table 7.6) were applied the four test images. Initially only the ASF (see section 4.6.2) structure was tested using 8nn connectivity, which was done for two reasons. Firstly this was done to see if the attributes could be psychovisually tuned and provide an improvement in compression and secondly to see if the number of comparisons could be reduced, thus shortening the testing period.

Several psychovisual evaluations were undertaken to determine the maximum value that can be used for each attribute, filter structure and connectivity while ensuring that the preprocessing is visually lossless. A range of observers were shown image pairs (original and preprocessed) with the processed image filtered to the attribute values shown in Table 7.14. A total of 428 comparisons were made with the visual tests to determine the visually lossless attribute values with an average 6.4 observers for each comparison. A comparison consists of 2 images, the original and the filtered with the four images used being filtered identically, so one set of results for a particular filter setting, consists of four images, which then gives an average of 25.6 observations for each filter setting. The image pairs are selected at random and the same un-impaired image was never shown successively. In addition the order within each image pair was selected randomly to remove any observer bias.

Area Limit	For 8nn ASF	5, 8, 12, 15, 18, 22, 25, 28, 32, 35, 38, 42, 45, 48, 52
	For All others	2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24
Power Limit		500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000, 6500, 7000

Table 7.14: Attribute limits used for the test.

One way of choosing a visually lossless limit is to select the smallest attribute value found to be acceptable over all of the images, which would guarantee all of the test images (Barbara, Barbara 2, Boats and Goldhill) would be visually lossless. However, this is very dependant upon the test data. For example, an image with a star-field in the background would inevitably require a very small area attribute, otherwise the stars would disappear. Hence whilst this method would guarantee the images are visually lossless, little gain could be achieved, especially considering this scenario. A better method is to take all of the results for all images using a particular configuration into account. Hence the average of all of the data could be used and a straight line using the Least Mean Squares (LMS) fitted to the data.

The results for the test images (Barbara, Barbara 2, Boats and Goldhill) are analysed using the MOS (see section 3.2.2.4), as shown on the plots, see Figures 7.45 – 7.52, as points. The MOS is taken over all four test images for the same configuration. A straight line is then fitted to the data, taking the first measured point before this line cuts across an MOS of –1 to be used as the visually lossless value. An MOS of -1 or less is considered to be a visual degradation and any value +1 or greater is considered an improvement upon the original image, which is a standard set out by the ITU [36]. Using a straight line fit will give an attribute value that can be considered visually lossless for the majority of images. Figures 7.45 – 7.52 show the results for the four combinations of the morphological filter where the points show the MOS calculated from all four images and the solid line shows the LMS fit to the data with the error bars showing the 95% confidence interval for the results taken at each point. All of the results show the trend that the MOS decreases as the attribute value increases.

Figure 7.45 shows the area attribute results using the AF filter structure (see section 4.6.2) and 4nn connectivity, which show that after an area size of 12, the majority of results have an MOS of –1 or lower indicating that the filtering can be observed visually. The linear fit to the data confirms this as it crosses the MOS value of –1 between an area size of 12 and 14. Hence the area size of 12 is used as the visually lossless attribute value. In addition, the error bars, which show the 95% confidence interval, show that only at an area size of 22, the bars are completely under the MOS value of –1. Thus indicating that there is a 95% chance that people will see the degradation. However, at an area size of 12 part of the bar is still above the MOS value of –1 indicating that there is still a probability that observers cannot see the change in quality. Changing the connectivity to 8nn as shown in Figure 7.46, shows that the LMS fit crosses the MOS value of –1 between an area size of 22 and 24. Hence the visually lossless area size is set to 22 when using 8nn connectivity, the AF filter structure and the are attribute. The results do show that using 8nn connectivity for the AF filter structure tends to produce a more visually acceptable output using higher area size values. It is also apparent that the 95% confidence interval never goes completely below the –1 MOS value.

The results for using 4nn connectivity, the area attribute and the ASF filter structure (see section 4.6.2) are shown in Figure 7.47 and are very similar to those produced using the AF filter structure (see Figure 7.45). The degradation is not visually noticeable until after an area size of 12 and so the visually lossless attribute value of 12 is used. In addition the 95% confidence interval is only completely below the –1 MOS value for an area size of 20 and 22, so the attribute value could actually be increased to 18 and still retain a strong chance that observers would not be able to see a change in the quality. Figure 7.48 shows the result of changing this filter to use 8nn connectivity. As can be seen, the LMS fit crosses the –1 MOS line between an area size of 15 and 18. Hence the visually lossless value for area size for this filter configuration is set to 15. This is lower than the previous configuration that used 8nn connectivity, which used an area size of 22 but is still higher than either of the methods using 4nn connectivity. Figure 7.49 shows the morphological filter using 4nn connectivity, the AF filter structure and the power attribute in place of the area. This shows that at a power of 2500, or just after, the degradation in the image can be seen visually. Since a power of 2000 is so close to the MOS value of –1, the conservative approach is taken and so the visually lossless limit is set to be the measured value before this, 2000 to ensure less visible degradation in the filtered image. As with the area attribute, changing the filter to use 8nn connectivity shows that the attribute value is larger than with 4nn connectivity (see Figure 7.50). This shows a visual degradation appears just prior to the power value of 3500 and so the visually lossless power value of 3000 is used.

Using the ASF filter structure in combination with 4nn connectivity and the power attribute (see Figure 7.51) shows a slightly higher attribute value can be used, 2500, than had the AF filter structure been used, which is due to the fact that the ASF works iteratively remove bright objects at a given attribute, then dark objects and then increasing the attribute and repeating the procedure. The AF by contrast simply removes all of the light components to a given attribute and then repeats the operation with the dark components, which is more visually noticeable. The attribute value can be increased further still by using the 8nn connectivity with the ASF filter structure. The LMS fit crosses the MOS value of –1 between a power value of 5000 and 5500. However, since the line is so close to an MOS of –1 at a power value of 5000, the conservative approach is taken and the value of 4500 is used as the visually lossless attribute value.

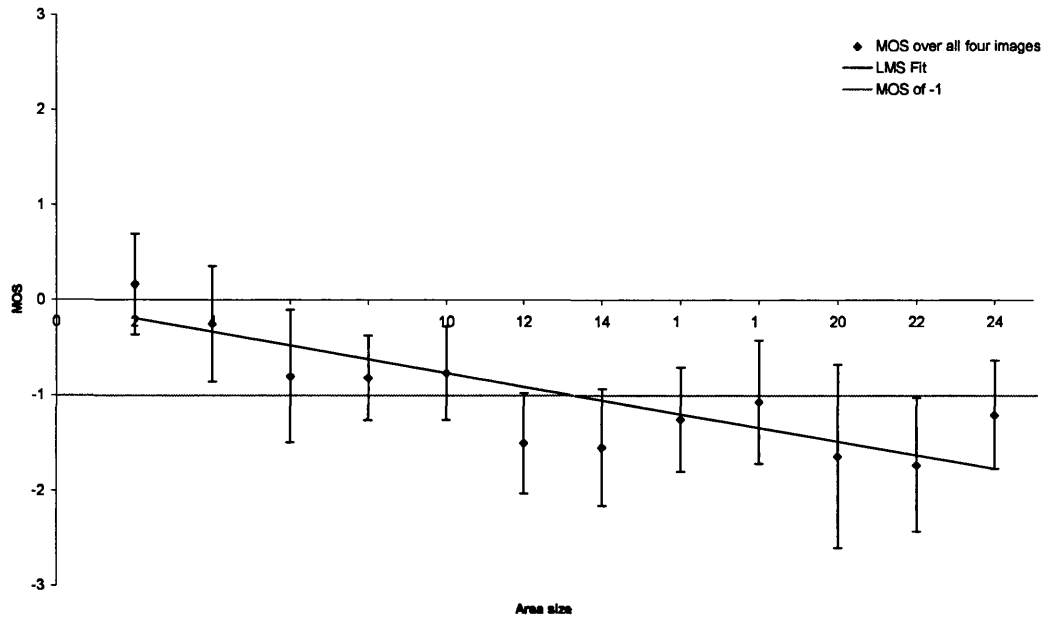


Figure 7.45: Psychovisual test results for 4nn connectivity using the AF filtering structure and the area attribute. The error bars show the 95% confidence interval.

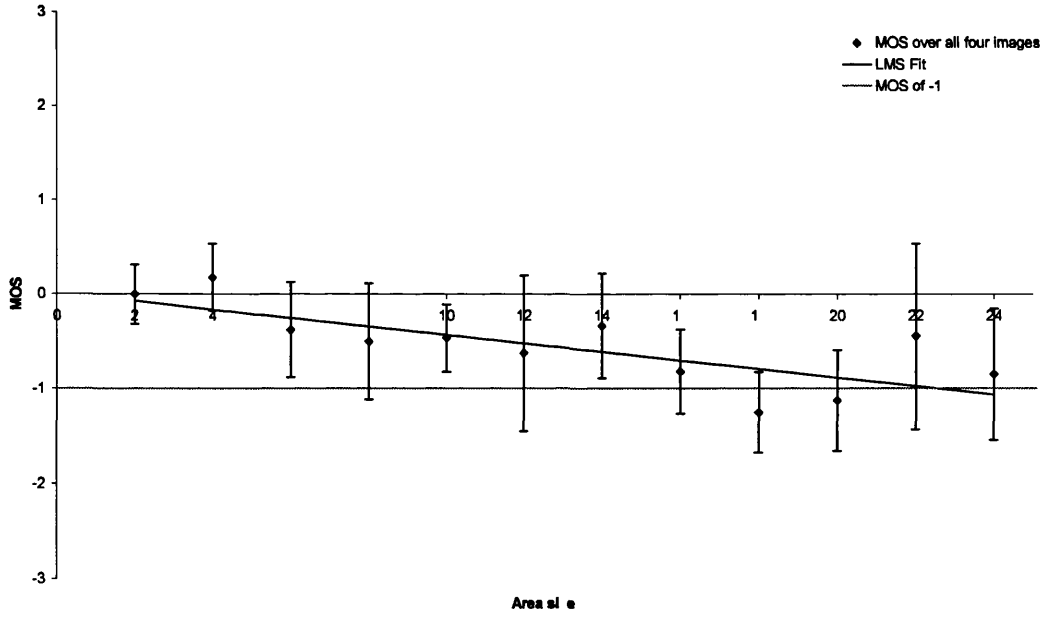


Figure 7.46: Psychovisual test results for 8nn connectivity using the AF filtering structure and the area attribute. The error bars show the 95% confidence interval.

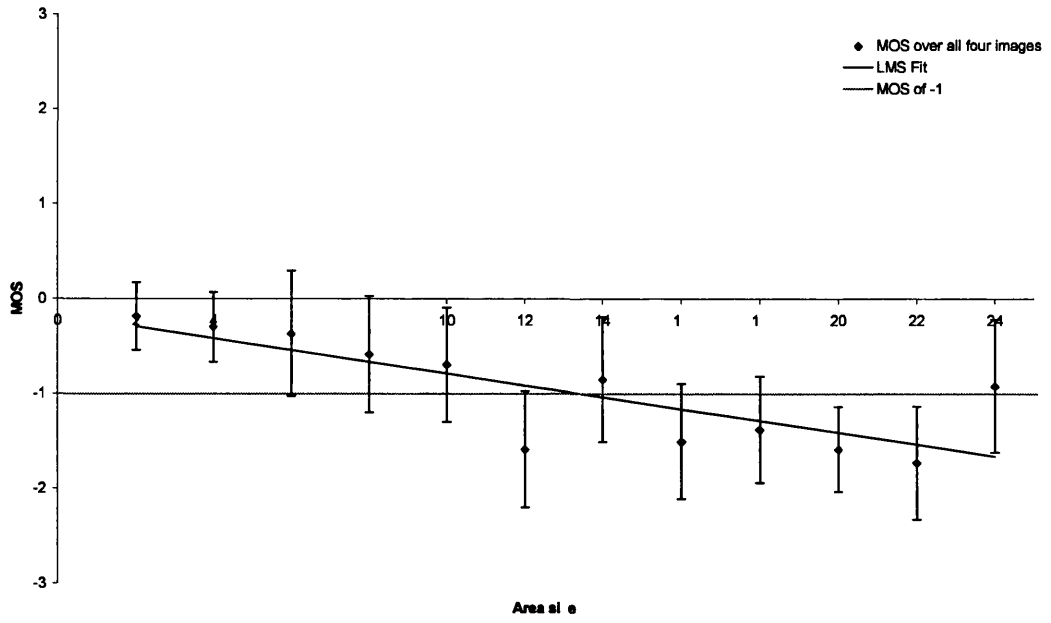


Figure 7.47: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the area attribute. The error bars show the 95% confidence interval.

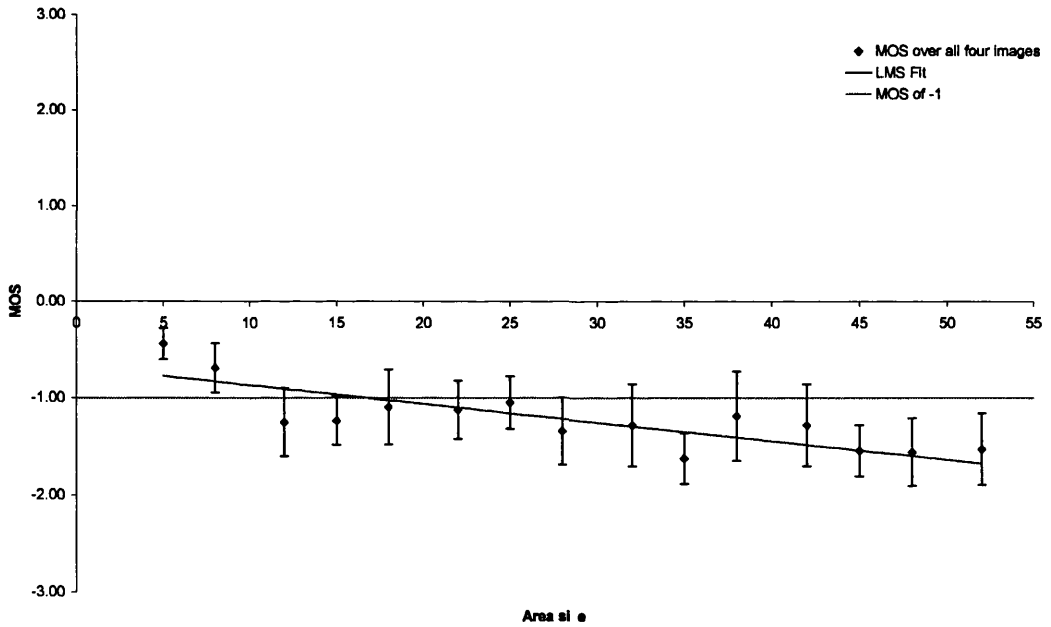


Figure 7.48: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the area attribute. The error bars show the 95% confidence interval.

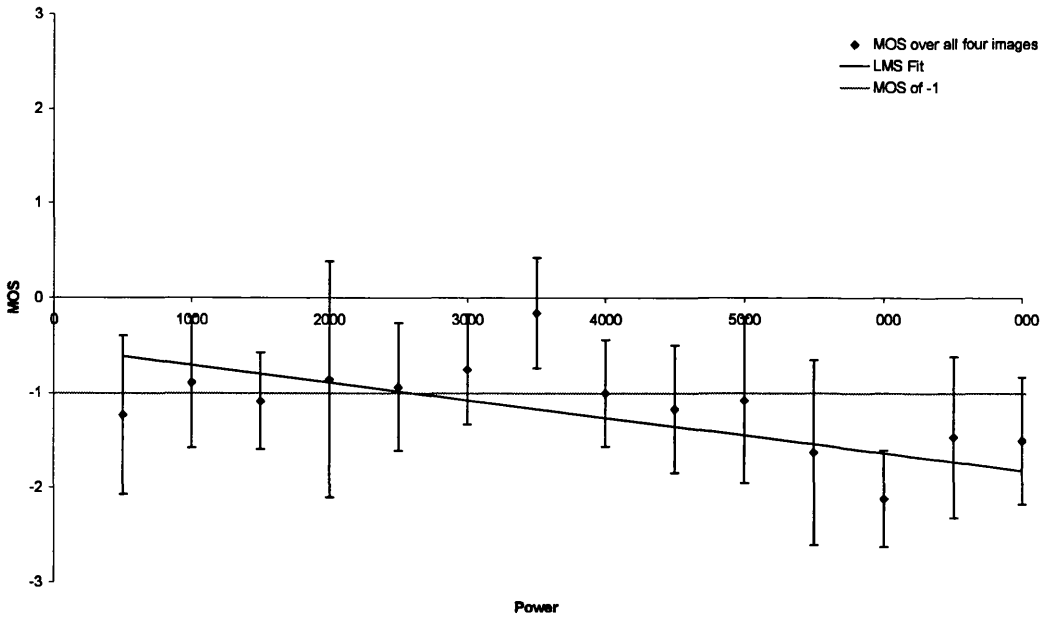


Figure 7.49: Psychovisual test results for 4nn connectivity using the AF filtering structure and the power attribute. The error bars show the 95% confidence interval.

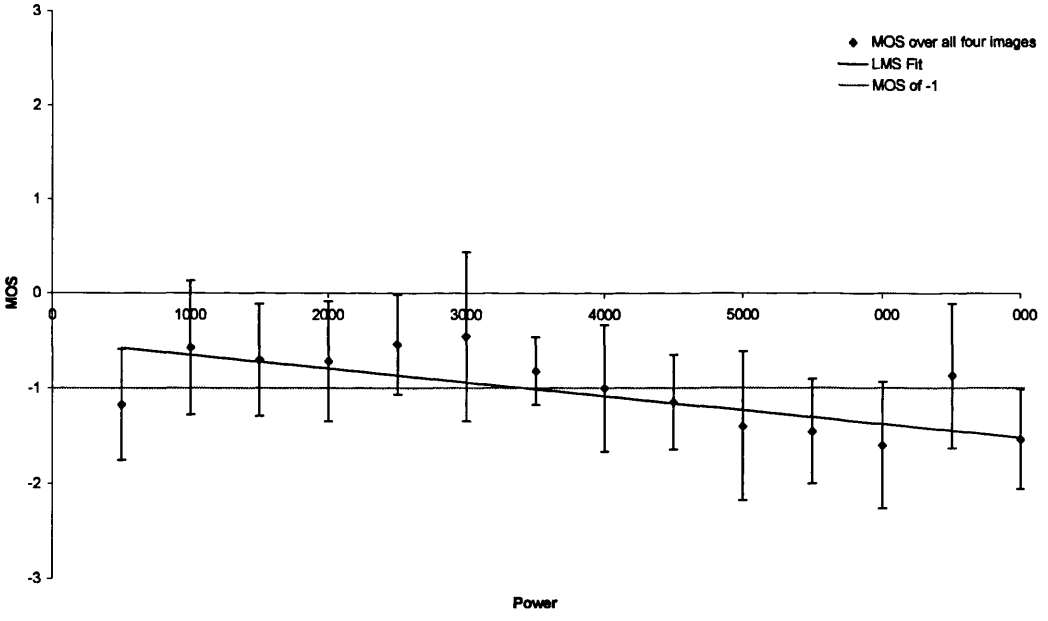


Figure 7.50: Psychovisual test results for 8nn connectivity using the AF filtering structure and the power attribute. The error bars show the 95% confidence interval.

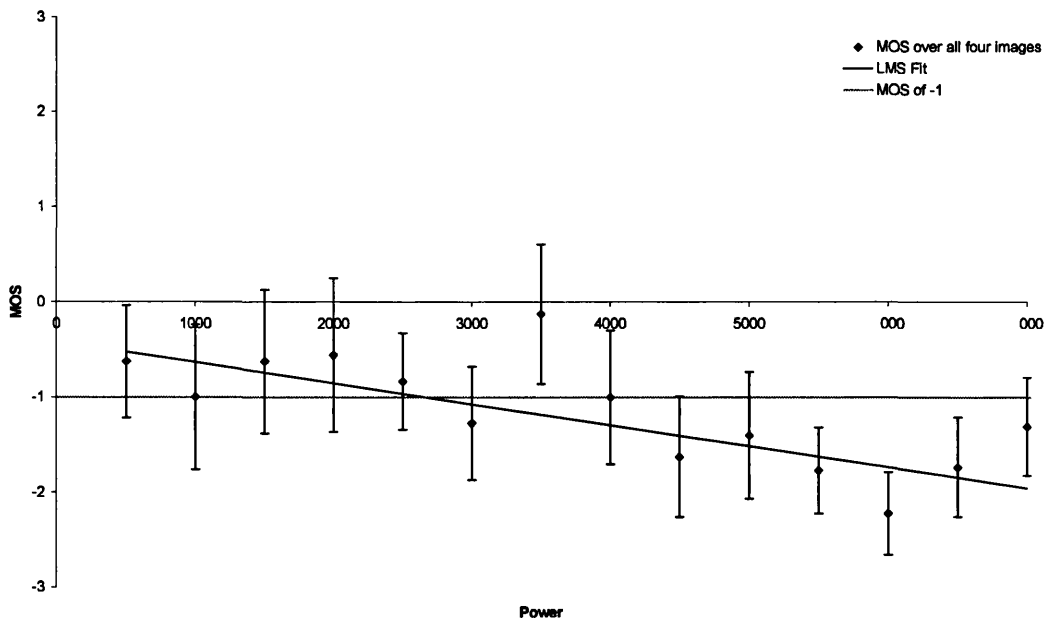


Figure 7.51: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the power attribute. The error bars show the 95% confidence interval.

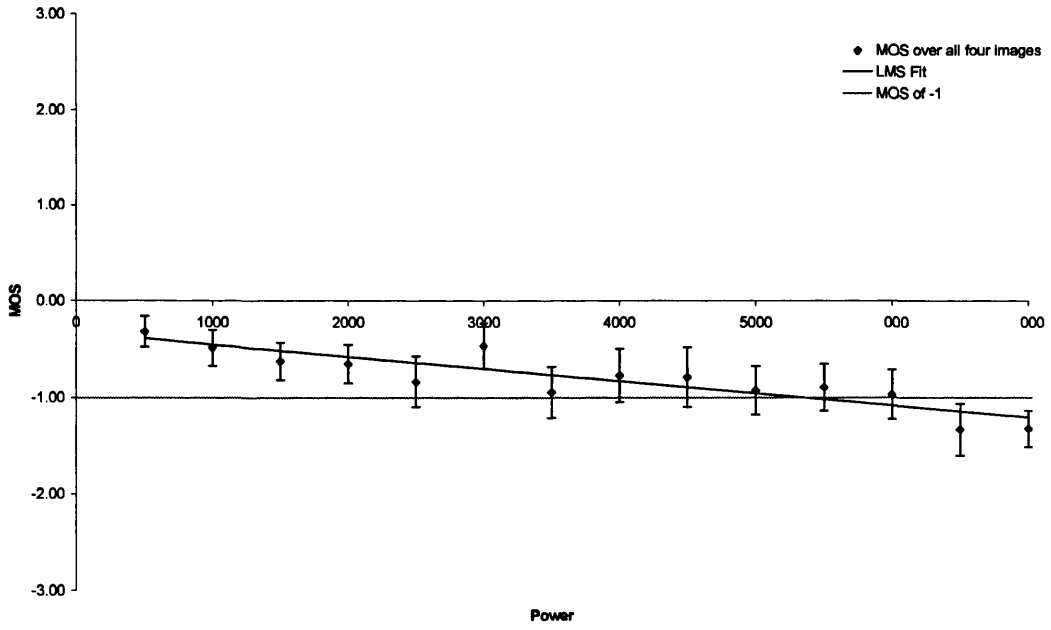


Figure 7.52: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the power attribute. The error bars show the 95% confidence interval.

Table 7.15 shows the attribute values that were determined to be visually lossless. The ASF attribute values can be seen to be the same value or lower when using the area attribute than the AF attribute value. The power attribute values show the opposite though. To give an indication of what the filters actually do to the images, the area and power attributes have been used to filter the Barbara image (see Figure 7.53) using 4nn connectivity and the AF filter structure as shown in Figures 7.54 and 7.55 respectively. Figure 7.54 shows the Barbara image filtered using the area attribute with values of 2, 12 and 24. A value of 2 (see Figure 7.54a) corresponds to the lowest amount of filtering shown in the visual tests and hence the least distorted. Few of the observers could see any degradation in this image. An area value of 12 (see Figure 7.54b) corresponds to the visually lossless limit that was estimated using the linear LMS fit. At this point some of the observers can see a difference whilst overall most cannot. Figure 7.54c shows the area of value 24, which is the highest amount of filtering used with this configuration in the test where most observers could see a difference.

Attribute	Connectivity	Filter Structure and attribute value	
		AF	ASF
Area	4nn	12	12
	8nn	22	15
Power	4nn	2000	2500
	8nn	3000	4500

Table 7.15: Attribute values that were determined to be visually lossless.

Figure 7.55 shows the Barbara image (the original Barbara image is shown in Figure 7.53) filtered using 4nn connectivity, the AF filter structure and the power attribute with values of 500, 2000 and 7000. A power value of 500 (see Figure 7.55a) corresponds to the least amount of filtering used with the power attribute in the tests. Few observers were able to see degradation at this level whilst at the highest level of filtering, a power value of 7000 (see Figure 7.55c), the majority of observers were easily able to see a degradation in the image quality. The visually lossless value for this filter was determined to be 2000 an is shown in Figure 7.55b. Both attributes using the lossless values show some degradation, although it is generally not noticed by non-image processing experts and hence is acceptable.

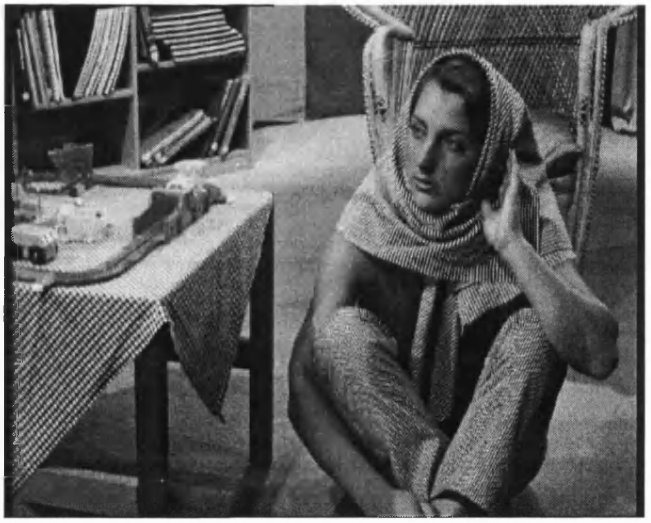
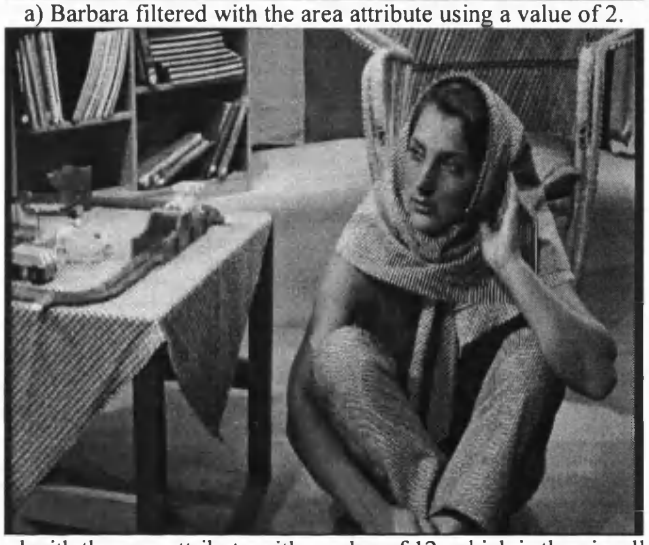
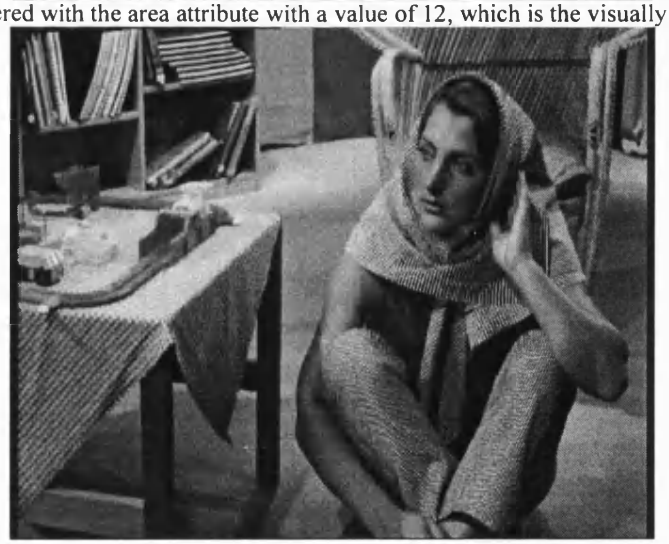


Figure 7.53: The original 8bit greyscale, 720 x 576 Barbara image.

a) Barbara filtered with the area attribute using a value of 2.



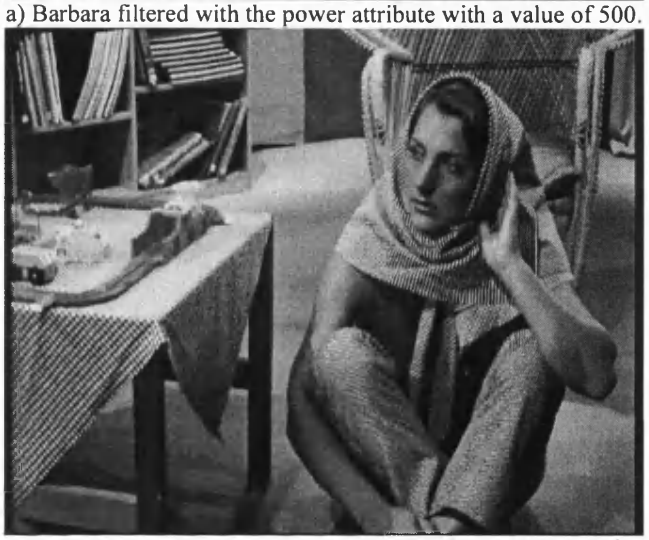
b) Barbara filtered with the area attribute with a value of 12, which is the visually lossless value.



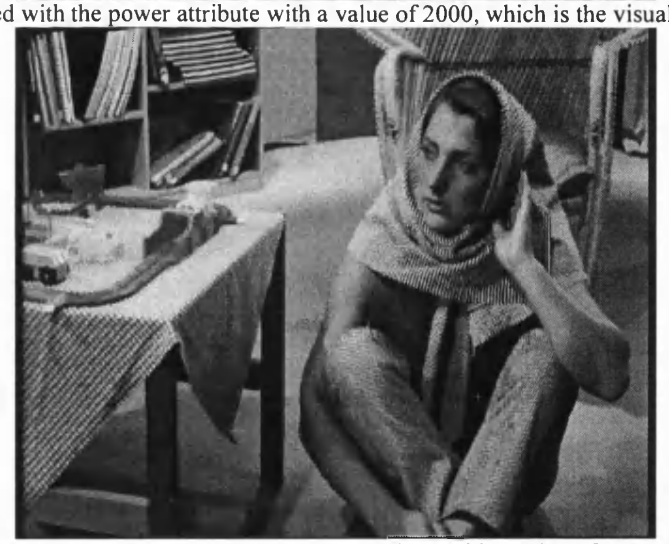
c) Barbara filtered with the area attribute with a value of 24.

Figure 7.54: The Barbara image filtered using the AF filter structure, 4nn connectivity and the area attribute.

a) Barbara filtered with the power attribute with a value of 500.



b) Barbara filtered with the power attribute with a value of 2000, which is the visually lossless value.



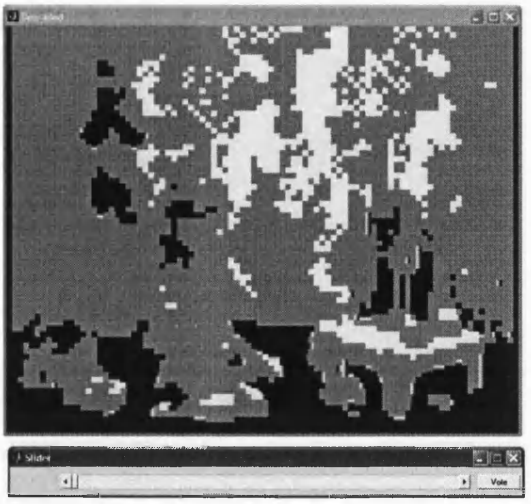
c) Barbara filtered with the power attribute with a value of 7000.

Figure 7.55: The Barbara image filtered using the AF filter structure, 4nn connectivity and the power attribute.

7.2.3 CODEC Output Evaluation

Section 7.2.1 showed that the CODEC performance increases with attribute value. Although section 7.2.2 showed that the output of the CODEC is visually lossless, meaning that an observer cannot see the difference between the output and the original input, it is still numerically lossy, meaning that the although the output looks identical to the input, numerically they may not be. Thus the CODEC will handle it in a different way, which may introduce distortions such as visible artefacts on the decoded output. To determine if this is the case, a further set of psychovisual tests are conducted. There are several ways in which this test may be conducted, the first of which would be used to determine at which RMSE the output of the CODEC's using the preprocessed images as inputs was visually lossless in comparison with the output produced by the CODEC using the original image. This is restrictive as a suitable quality for the reference image and range of qualities for the degraded images needs to be selected.

The chosen method works around this by compressing and decompressing the images after being filtered using the visually lossless attribute values to a set compression ratio, which is 20:1 (or 0.4bpp). This is then shown to the observer as is the degraded image, which is simply the original unfiltered image having been compressed and decompressed. However, rather than setting this to a few ranges of quality or compression ratios, the observer is also given a slider bar as shown in Figure 7.56. With the slider bar fully to the left, the highest compression ratio of 80:1 (or 0.1bpp), or lowest quality is shown. As the slider is moved to the left, the compression ratio decreases and thus quality increases until the slider is fully over to the left, which corresponds to a compression ratio of 1:1 (or 8bpp). The observer is asked to move the slider to the left and stop at the point where they think the quality of this image matches the original reference image they were shown. A total of 7 observers were used in this evaluation. Results from this test are also analysed in a different way to the previous tests. The reference image, that is the image that was filtered and then compressed to 20:1 (or 0.4bpp) is compared to the compression ratio of the original unfiltered image. The compression ratio of the original image at the point where the observers thought both images had the same quality was then recorded. Results were then analysed by counting how many users thought that the same quality was achieved by using a lower compression ratio on the original, the same compression ratio or a higher compression ratio as shown in Table 7.16. If the original image needed a lower compression ratio to give the same visual quality as the filtered image, then this shows that the filtering improves the compressibility of an image whilst retaining the visual quality. However if a higher compression ratio is required, then the visual quality has been degraded via the filtering process.



a) The starting screen, a compression ratio of 80:1 (or 0.1bpp).

b) An intermediate shoot as the observer moves the slider decreasing the compression ratio and increasing quality.

c) The slider fully across corresponding to the best quality and a 1:1 compression ratio (or 8bpp)

Figure 7.56: The setup of the second visual test using a slider bar to control the compression ratio/quality.

Table 7.16 shows the full set of results for all filter combinations and Table 7.17 shows the average compression ratio chosen by each observer that lays in the highest range indicated by Table 7.16. The first table, Table 7.16, shows that for both the JPEG and JPEG 2000 CODEC's, most of the observers, 71% and 54% respectively, thought that the compression ratio required from the non-filtered image needed to be lower than 20:1 to achieve the same quality as the filtered image compressed to a ratio of 20:1, indicating that the quality remains the same for the filtered image, whilst obtaining a much higher compression ratio. For example using 4nn connectivity, the AF filter structure, the area attribute and the JPEG CODEC, 75% of the observers thought that the original, un-filtered image needed a compression ratio lower than 20:1 to get the same quality as that of the filtered image.

Filter combination used for the reference image.			Percentage of observers who think that the original image compressed to x:1 looks the same as the filtered compressed to 20:1 (0.4bpp)					
			JPEG			JPEG 2000		
Connectivity	Attribute	Filter Structure	<20:1	=20:1	>20:1	<20:1	=20:1	>20:1
4nn	Area	AF	**75%**	4%	21%	**57%**	7%	36%
		ASF	**64%**	7%	29%	**61%**	0%	39%
	Power	AF	**71%**	11%	18%	**57%**	0%	43%
		ASF	**75%**	7%	18%	**68%**	3%	29%
8nn	Area	AF	**79%**	0%	21%	**54%**	0%	46%
		ASF	**61%**	14%	25%	**46%**	8%	46%
	Power	AF	**68%**	7%	25%	46%	4%	**50%**
		ASF	**79%**	0%	21%	**46%**	8%	**46%**
Total over all the results			**71%**	6%	22%	**54%**	4%	42%

Table 7.16: Results to show how many observers thought what compression ratios compared to the filtered at 20:1 (0.4bpp). The shaded results show the best outputs.

Filter combination used for the reference image.			Average compression ratio for the highest percentage	
Connectivity	Attribute	Filter Structure	JPEG	JPEG 2000
4nn	Area	AF	15.0	**15.2**
		ASF	14.2	15.5
	Power	AF	**13.9**	15.6
		ASF	15.0	16.38
8nn	Area	AF	14.2	16.2
		ASF	15.0	16.3
	Power	AF	15.0	24.7
		ASF	14.7	15.8 / 24.7
Average for all results under 20:1			14.6	15.9

Table 7.17: The average compression ratio laying within the most frequently chosen range of Table 7.16.

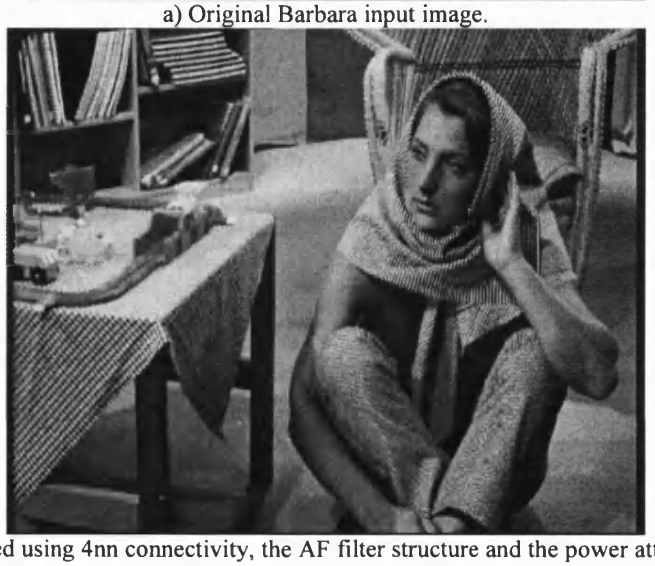
Thus from Table 7.16 it can be concluded that the filtered images do increase the compressibility of the images whilst maintaining the visual quality. The only exception is that of the filter using 8nn connectivity, the AF filter structure with the power attribute and the JPEG 2000 CODEC. This shows that most users though that a compression ratio of greater than 20:1 was required from the unfiltered image to make it look the same as the filtered and hence showing that the filtered image decreases the performance of the CODEC. However 8% of the observers also said that a difference could not be

seen so this could be considered a tie. In addition when the filter structure is changed to ASF, it is an even split between the results. The JPEG 2000 show that less observers chose a compression ratio lower than 20:1 than was chosen for JPEG. This would indicate that JPEG 2000 is a much better CODEC in terms of producing less visible degradations, which is a result of using the wavelet transform.

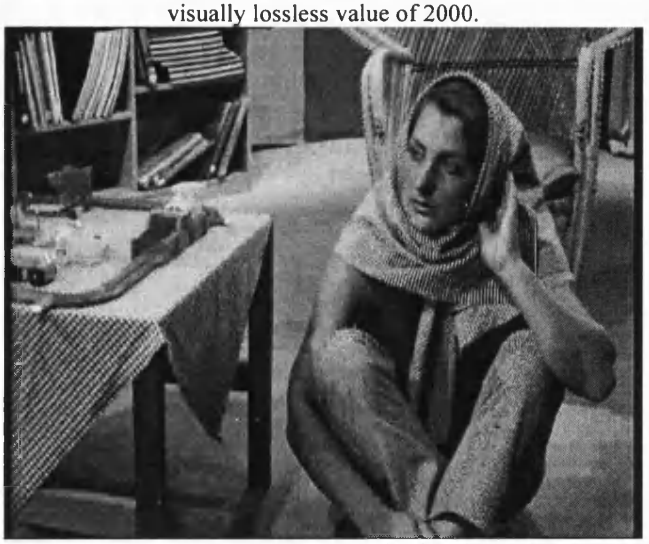
To give an indication of the increase in compression ratio, the average compression ratio chosen by each observer is averaged so long as it lays in the best region as indicated by Table 7.16. Hence using the 4nn connectivity, the AF filter structure, the area attribute and the JPEG CODEC, since 75% of the observers thought that a compression ratio of less than 20:1 was required from the unfiltered image, only the observer's results that lay in this range are averaged. From Table 7.17 it can be seen that for this configuration, the average compression ratio was 15:1, thus showing that the filtered image increases the compression by 33% to 20:1 and still retains the visual quality. The results also show that the 8nn connectivity results for both the JPEG and JPEG 2000 CODEC's give the same or higher compression ratio than the 4nn connectivity, indicating that the 4nn connectivity filters produce a more compressible image.

The best outputs of the two CODEC's are not produced using the same filters, but both of the best outputs use 4nn connectivity and the AF filter structure, but the JPEG CODEC uses the power attribute whilst the JPEG 2000 uses the area attribute. Figure 7.57 shows the original Barbara image, the filtered version using the 4nn connectivity, the AF filter structure and the power attribute with a value of 2000, which was the attribute value that was determined to be visually lossless (see section 7.2.2). In addition the image that is used as a reference is shown, which is the filtered version compressed and decompressed using the JPEG CODEC to a compression ratio of 20:1 (0.4bpp). The degraded images, that is the original image compressed to different levels is shown in Figure 7.58. This shows the lowest compression ratio that was voted for, which was 3:1 (2.7bpp). This means that the original unfiltered image compressed and decompressed to a ratio of 3:1 was seen to look the same as the filtered version compressed and decompressed to a ratio of 20:1 (0.4bpp) by at least 1 observer. Since 71% (see Table 7.16) of the observers thought that the degraded image with a compression ratio lower than 20:1 looked the same as the reference image and because several of the compression ratios had the same number of votes, the average compression ratio of those voted for under a ratio of 20:1 are averaged together and shown in Figure 7.58b. The last image, Figure 7.58c, shows the highest compression ratio, 27:1 (0.3bpp), which was voted for. It is hard to see the difference between the two lower ratios, Figure 7.58a and b, and the reference, Figure 7.57c. However the highest compression ratio, it is clearly visible that the reference looks far superior.

a) Original Barbara input image.



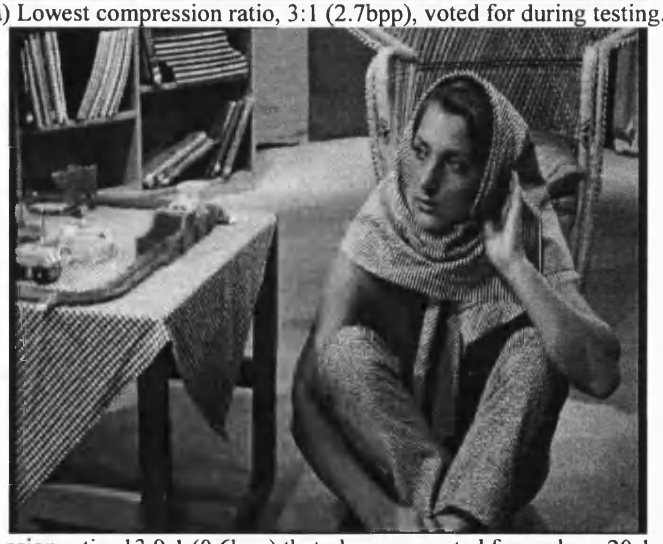
b) Image filtered using 4nn connectivity, the AF filter structure and the power attribute with the visually lossless value of 2000.



c) The filtered image (b) compressed and decompressed using the JPEG CODEC to a ratio of 20:1 (0.4bpp). This is used as a reference image in the visual test.

Figure 7.57: The input, filtered and reference images (8bit greyscale, 720 x 576) used in testing.

a) Lowest compression ratio, 3:1 (2.7bpp), voted for during testing.

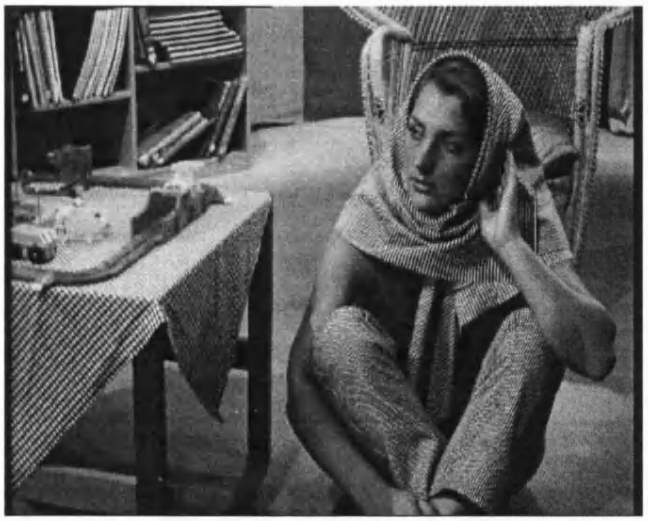


b) Average compression ratio, 13.9:1 (0.6bpp) that observers voted for under a 20:1 compression ratio.

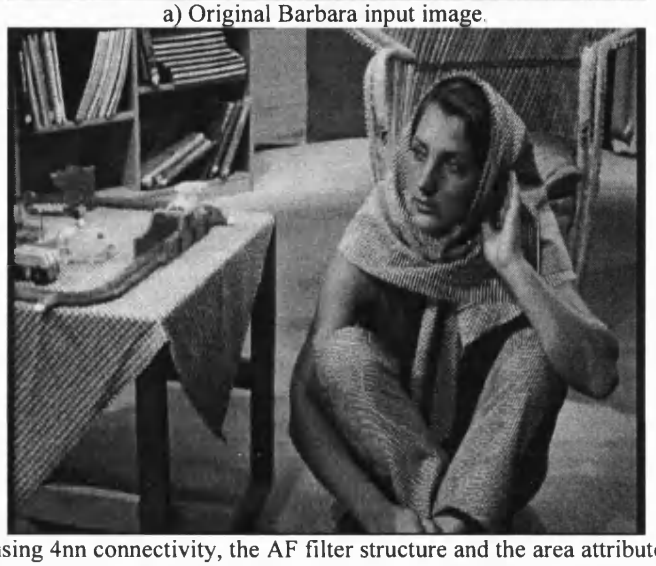
c) Highest compression ratio, 27:1 (0.3bpp) that observers voted for.

Figure 7.58: The best JPEG result output, showing the lowest compression ratio voted for, the average of those laying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for.

Figure 7.59 shows the original unfiltered Barbara image, the filtered version using the 4nn connectivity, the AF filter structure and the area attribute with a value of 12, which was the value determined to be visually lossless (see section 7.2.2). Even for an expert, it is difficult to see differences between these two images. In addition the compressed and decompressed image using the JPEG 2000 CODEC and a compression ratio of 20:1 (0.4bpp) is shown. This last image is used as the reference image for the visual testing. The degraded images voted for are shown in Figure 7.60. This shows the lowest compression ratio voted for, 8: (1bpp). Only 57% of the observers thought that a compression ratio of less than 20:1 looked the same as the reference image. However, since this is still the majority, the average of the compression ratios voted for is also calculated, which is found to be 15.2:1 (0.5bpp). It is very hard to see a difference between these two images and the reference (see Figure 7.59a), but using the highest compression ratio voted for, 31:1 (0.3bpp) it becomes much easier to see that the filtered version looks much better.



a) Original Barbara input image.



b) Image filtered using 4nn connectivity, the AF filter structure and the area attribute with the visually lossless value of 12.

c) The filtered image (b) compressed and decompressed using the JPEG 2000 CODEC to a ratio of 20:1 (0.4bpp). This is used as a reference image in the visual test.

Figure 7.59: The input, filtered and reference images (8bit greyscale, 720 x 576) used in testing.

a) Lowest compression ratio, 8 (1bpp), voted for during testing.

b) Average compression ratio, 15.2:1 (0.5bpp) that observers voted for under a 20:1 compression ratio.

c) Highest compression ratio, 31:1 (0.3bpp) that observers voted for.

Figure 7.60: The best JPEG 2000 result output, showing the lowest compression ratio voted for, the average of those laying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for.

7.2.4 Application to Unseen Data

This first test is to show if the visually lossless attribute values found in section 7.2.2 can be generalised to any image or if the values are specific to the images that were used to find them. The two images are first filtered using the same range of attribute values and filters as was in section 7.2.2 (see Table 7.18). A psychovisual test was conducted in exactly the same way, that is using the DSIS method (see section 3.2.2.1). A total of 6 observers evaluated each image, the results of which were analysed using the MOS and the 95% confidence intervals (see section 3.2.2.4). The results are shown in Figures 7.61 – 7.68, which show the linear LMS fit to the original training data (see section 7.2.2), the new results for the Blackboard and Girl images shown as points and a LMS fit to that data with 95% confidence intervals of the Blackboard and Girl results shown.

Area Limit	For 8nn ASF	5, 8, 12, 15, 18, 22, 25, 28, 32, 35, 38, 42, 45, 48, 52
	For All others	2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24
Power Limit		500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000, 6500, 7000

Table 7.18: Attribute limits used for the test.

With no exceptions, all of the new results show that the LMS fit to the data crosses the MOS value of –1 at the same or higher attribute value than the original data did (see Table 7.19) and that the original fit lays within or close to the 95% confidence interval of the new data. From Figures 7.61 – 7.68, it can be seen though, like the original data (see Figures 7.45 – 7.52) that some of the individual results have an MOS score lower than –1, but have the same general trend as the original data for the LMS fit. Thus this shows that the original visually lossless attribute values (see Table 7.15) hold for images that have not been used in the training cycle. However, generally higher attribute values have been shown to be achieved by the unseen data, for example using the area attribute with 4nn connectivity and the AF filter structure, the original lossless value was determined to be an area size of 12 (see Table 7.15) whereas the new data uses an attribute value of 20. This indicates that it may be possible to increase the value of the visually lossless attribute value by using more training data, but that the original estimated attribute values hold for unknown data.

Attribute	Connectivity	Filter Structure and attribute value	
		AF	ASF
Area	4nn	20	18
	8nn	24	25
Power	4nn	3000	3000
	8nn	4000	5500

Table 7.19: Attribute values that were determined to be visually lossless.

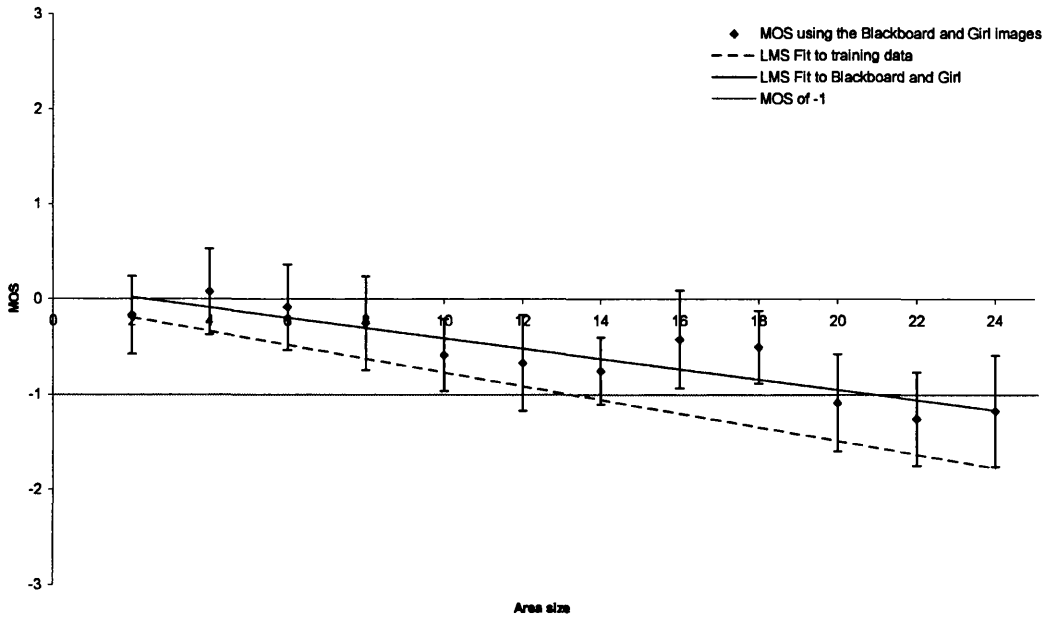


Figure 7.61: Psychovisual test results for 4nn connectivity using the AF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

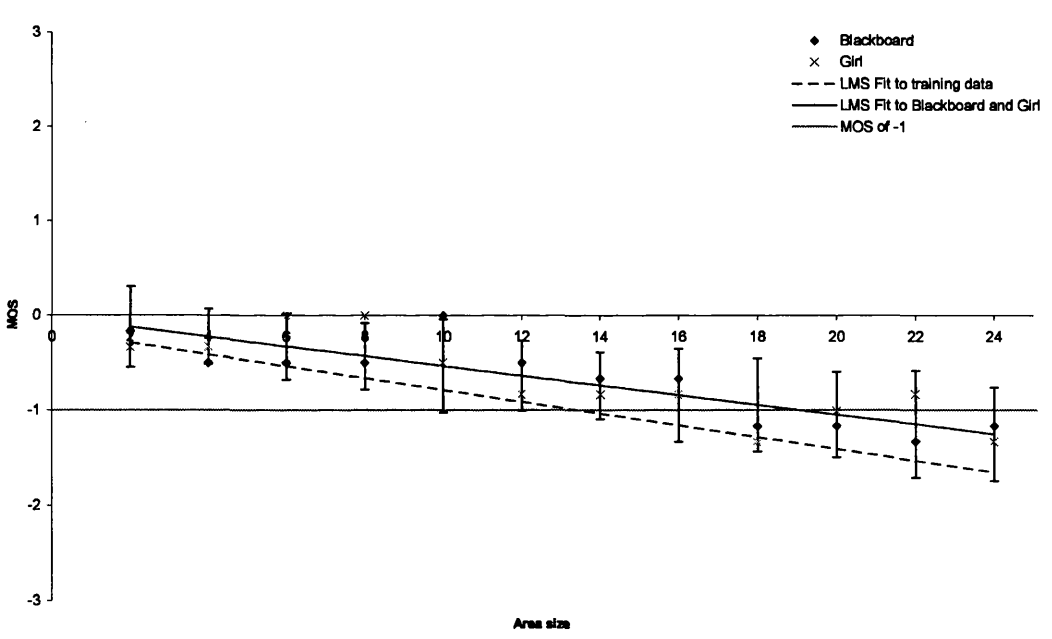


Figure 7.62: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

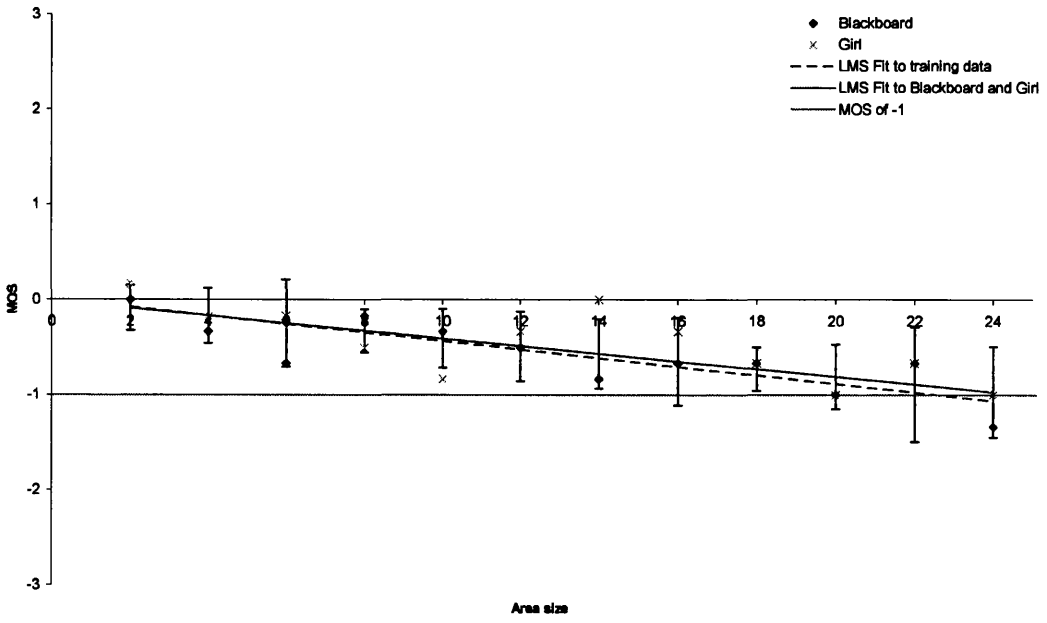


Figure 7.63: Psychovisual test results for 8nn connectivity using the AF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

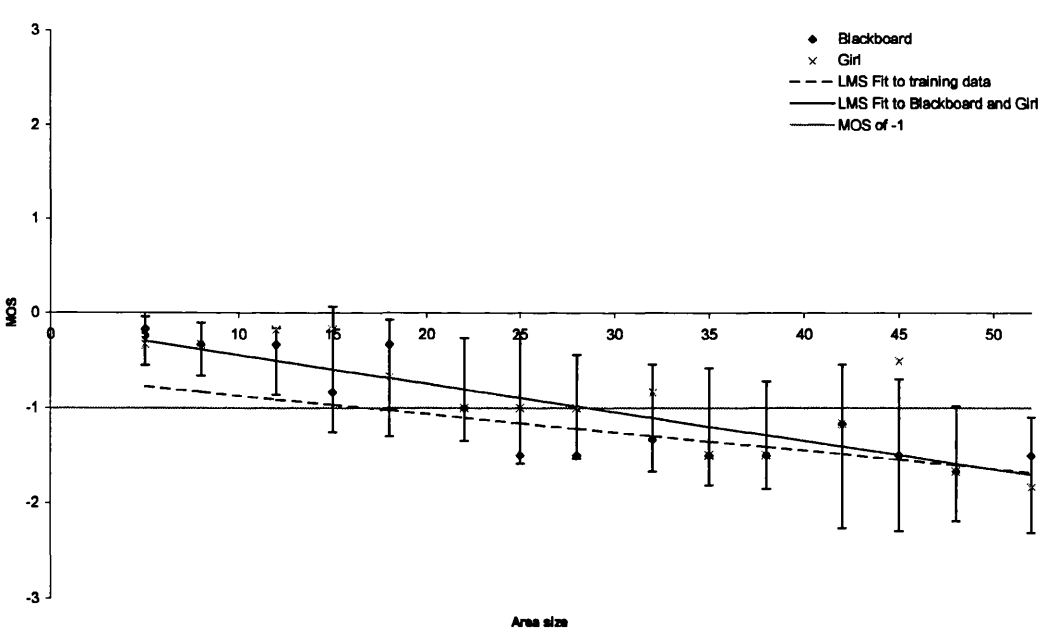


Figure 7.64: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the area attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

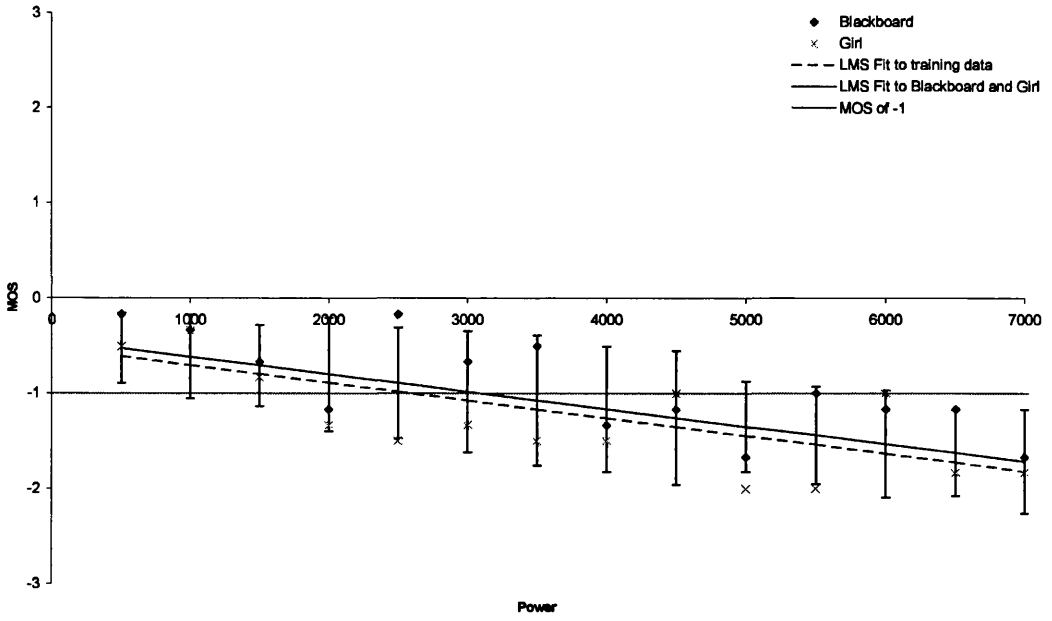


Figure 7.65: Psychovisual test results for 4nn connectivity using the AF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

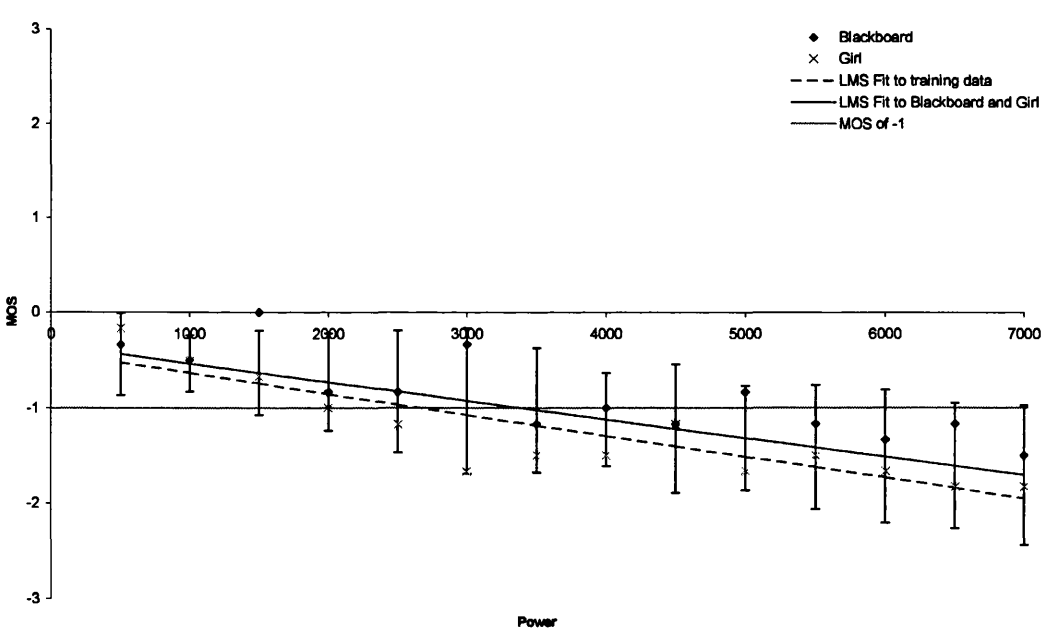


Figure 7.66: Psychovisual test results for 4nn connectivity using the ASF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

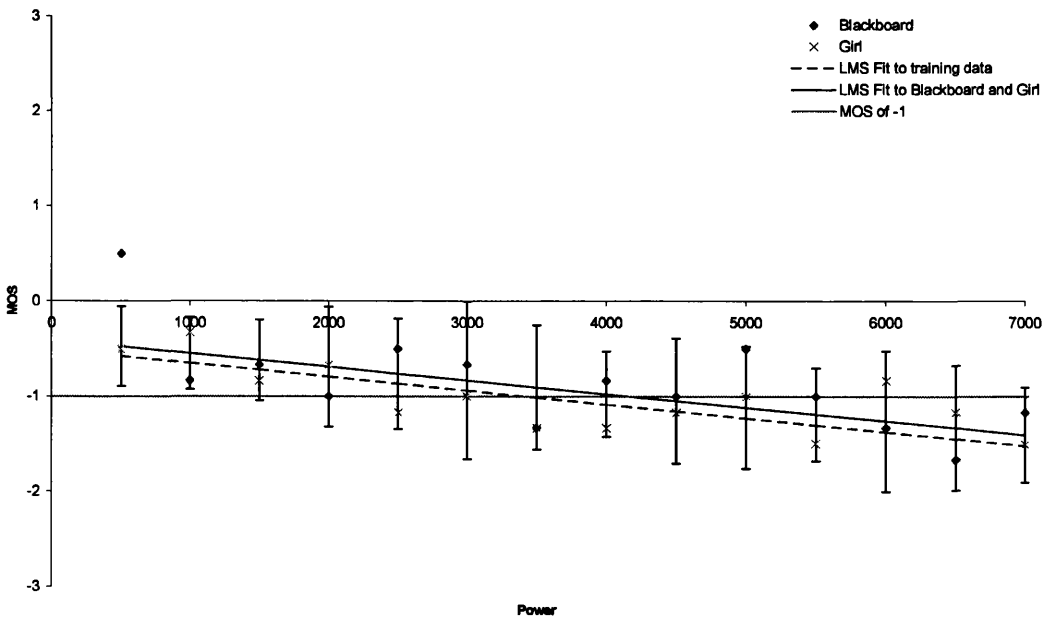


Figure 7.67: Psychovisual test results for 8nn connectivity using the AF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

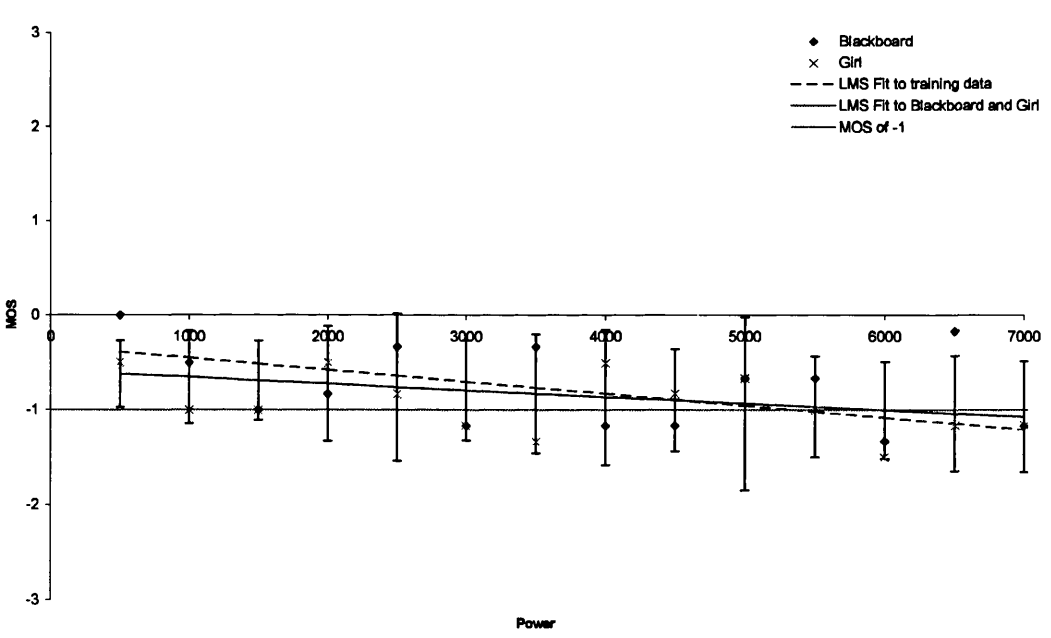


Figure 7.68: Psychovisual test results for 8nn connectivity using the ASF filtering structure and the power attribute. Error bars show the 95% confidence interval of the Blackboard and Girl results.

With the exception of the filter using 8nn connectivity, the ASF filter structure (see section 4.6.2) and the power attribute (see section 6.2.2.4) with a value of 1500 (see Figure 7.68), all of the results show that the 95% confidence interval, shown as the error bars, is either completely above the original LMS fir or that it lays within it. Hence this implies that the original estimates and testing can be generalised to any image, which is supported further by the fact that the points at which the LMS fit crosses the MOS value of –1 is the same or higher than it was for the training data. Thus the original estimates for the visually lossless attribute values (see Table 7.15) can be applied to any image and will be visually lossless.

To give an idea of how these images compare to the training data, some of the filtered results are shown in Figures 7.69, 7.70 and 7.71. Figure 7.69 shows the original unfiltered Blackboard image used for testing. The images filtered with the morphological filter consisting of 4nn connectivity, the AF filter structure and the area attribute with values of 2, 12 and 24 are shown in Figure 7.70. The first image shows the filtered image with an area size of 2, which corresponds to the least filtered image used during testing. Like the original, typically no difference can be seen with this little amount of filtering. The Second image shows the filter using the visually lossless area attribute, a value of 12, whereby some observers can see a degradation, but most cannot. However with the larger area size's, 24 for example, it is much easier to see degradation in the image quality. Figure 7.71 shows the images filtered using the morphological filter with the AF filter structure, 4nn connectivity and the power attribute with values of 500, 2000 and 7000. The first image shows the least amount of filtering used in the visual test with this configuration, which uses a power of 500. Very few observers were able to notice any degradation here. The visually lossless attribute value of 2000 is shown in the second image, where several observers were able to notice a change in the quality. The final image shows the highest degree of filtering used during testing. This used a power value of 7000, which has a very noticeable visual degradation in certain places, for example the wall is smoother than the output produced using the highest amount of filtering with the area attribute.

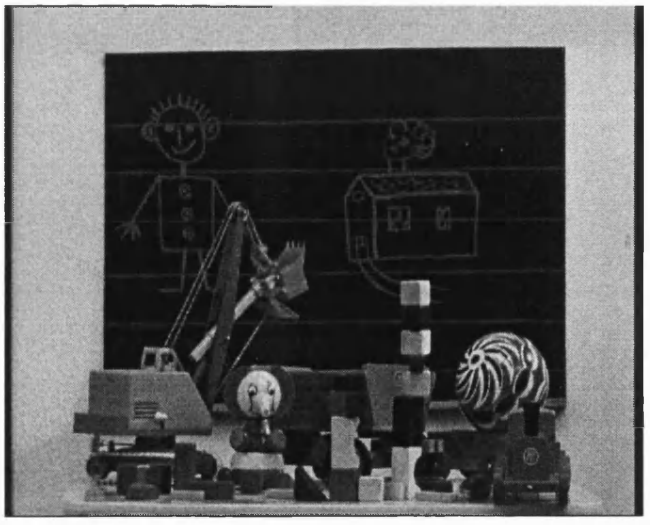
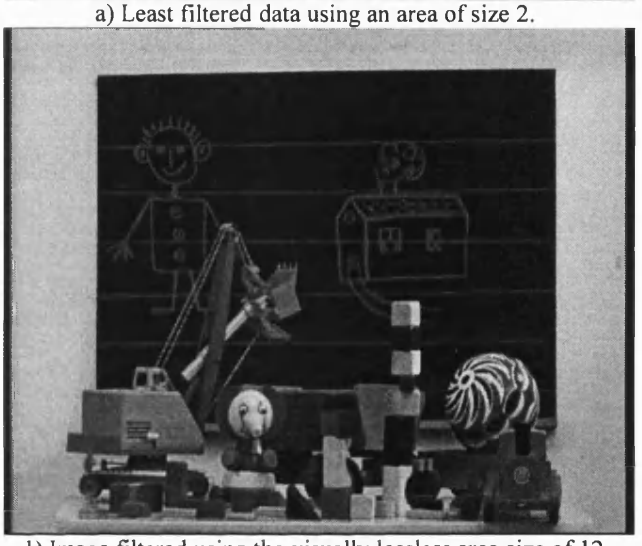


Figure 7.69: Input image, 720 x 576, 8bit greyscale Blackboard image, used to evaluate the visually lossless attribute values.

a) Least filtered data using an area of size 2.

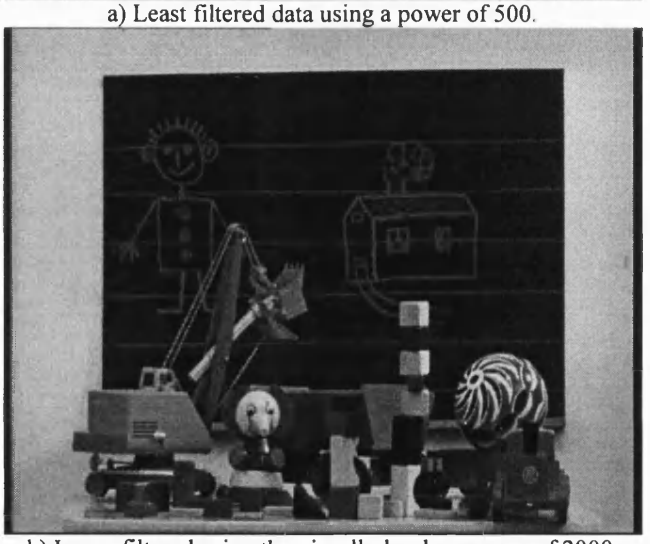


b) Image filtered using the visually lossless area size of 12.

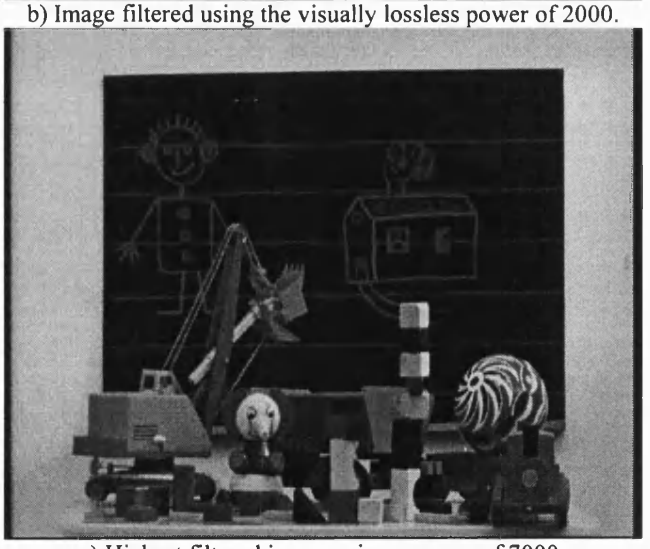
c) Highest filtered image using an area size of 24.

Figure 7.70: Filtered images used for testing using the area attribute, 4nn connectivity and the AF filter structure.

a) Least filtered data using a power of 500.



b) Image filtered using the visually lossless power of 2000.



c) Highest filtered image using a power of 7000.

Figure 7.71: Filtered images used for testing using the power attribute, 4nn connectivity and the AF filter structure.

7.2.4.1 CODEC Performance on Unseen Data

Although the previous section (section 7.2.4) showed that the visually lossless attribute values determined in section 7.2.2 can be applied to unseen data and still remain visually lossless, it is also required to prove that this will in addition still result in an improvement in the compressibility of the images. Hence the psychovisual test of section 7.2.3 is carried out using the Blackboard and Girl images filtered using the original visually lossless attribute values (see Table 7.15). Like before, the image is filtered using the visually lossless attribute values and then used as a reference image after being compressed with the JPEG or JPEG 2000 CODEC to a ratio of 20:1. The original image is then compressed and decompressed with the compression ratio being controlled by the observer. The observer is then asked to vote for the point where they think that the degraded image (the original after compression) that they think makes the two images look the same.

Table 7.20 shows the number of observers, as a percentage, who thought that the original image required a lower compression ratio (a higher bit-rate) than the image that had been filtered beforehand. There were a total of 7 observers who took part in this experiment. So for example using the morphological filter with 4nn connectivity, the power attribute and the AF filter structure, 79% of the observers thought that a compression ratio of less than 20:1 on the original image looked the same as the filtered image, which used the visually lossless attribute value of 2000, compressed to a ratio of 20:1 when using the JPEG CODEC. This indicates that the filtering improves the compressibility of the images whilst maintaining the visual quality. However if the JPEG 2000 CODEC is used to compress the output of the same filter, 57% of the observers thought that a compression ratio of greater than 20:1 was required from the original images to make it look the same as the filtered. Thus this case shows that filtering has degraded the compressibility of the images.

Filter combination used for the reference image.			Percentage of observers who think that the original image compressed to x:1 looks the same as the filtered compressed to 20:1					
Connectivity	Attribute	Filter Structure	JPEG			JPEG 2000		
			<20:1	=20:1	>20:1	<20:1	=20:1	>20:1
4nn	Area	AF	43%	7%	**50%**	43%	7%	**50%**
		ASF	**64%**	7%	29%	**50%**	7%	43%
	Power	AF	**79%**	0%	21%	36%	7%	**57%**
		ASF	**50%**	0%	**50%**	36%	7%	**57%**
8nn	Area	AF	**57%**	0%	43%	36%	14%	**50%**
		ASF	**43%**	14%	**43%**	**57%**	7%	36%
	Power	AF	**64%**	0%	36%	**50%**	7%	43%
		ASF	**43%**	21%	36%	21%	0%	**79%**
Total over all the results			**55%**	6%	39%	**41%**	7%	**52%**

Table 7.20: Results to show how many observers thought what compression ratios compared to the filtered at 20:1 (0.4bpp). The shaded results show the best outputs.

For JPEG, overall the observers thought that 55% of the time that a compression ratio lower than 20:1 was the same as the filtered images compressed to a ratio of 20:1, and that 6% of the time a compression ratio of 20:1 was thought to look the same. Thus for the majority of the cases, the observers showed that the filtered images produce images that can achieve either the same compression or higher whilst maintain the same visual quality as the original unfiltered images. However the JPEG 2000 CODEC shows the opposite. That is that overall, 52% of the observers said that a compression ratio higher than 20:1 was required to look the same as the filtered images compressed to a ratio of 20:1, thus showing that filtering impedes the CODEC's abilities to compress the images. The original training data (see Table 7.16) showed the opposite for the JPEG 2000 results. There are several possible reasons for this. The first is that the JPEG 2000 CODEC may act differently on the Blackboard and Girl images, producing more visible degradations than were visible on the training data. The second is that not enough images were used to get a true representation of the actual performance of the CODEC's. This could be for the training data, the evaluation images, or both.

To give an indication of the increase in compression achievable using this simplification method, the average compression ratio lying in the best output range is averaged as shown in Table 7.21. For example, using the filter composed of the AF filter structure, 4nn connectivity and the power attribute, since 79% of the observers said that a compression ratio lower than 20:1 was seen to be of the same quality as the filtered compressed to a ratio of 20:1, the average of all of the compression ratios voted for that are lower than 20:1 are taken. In this case an average compression ratio of 16.2:1 is found. This gives a 24% increase in compression. The most frequently voted for compression ratio is not used as several of them have the same number of votes and without further testing, the best value would not be known. Unlike the original CODEC evaluation, see section 7.2.3, the 8nn connectivity shows to produce the best increase in compression. This could be caused from several reasons. The first may be that content of the images are more suited to this type of connectivity. The second reason is that the attribute values need further optimisation, it maybe that if the attribute values are changed, the best filter may change. The final possibility is that more images may need to be evaluated and/or used in training the system to better optimise and evaluate the attributes.

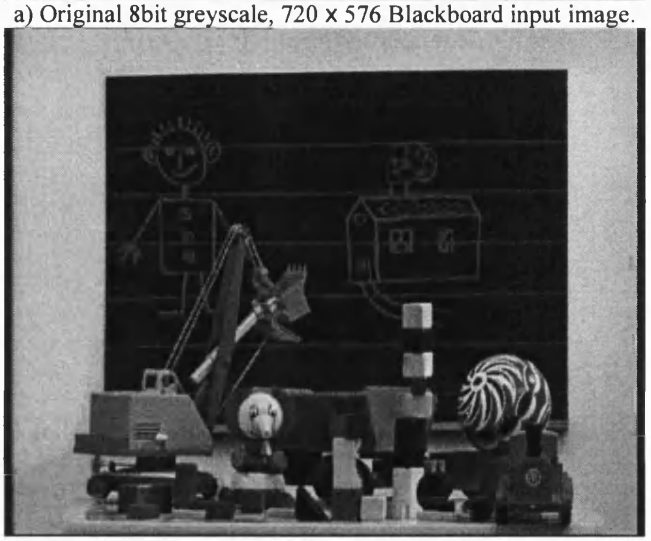
Filter combination used for the reference image.			Average compression ratio for the highest percentage (xx:1)	
Connectivity	Attribute	Filter Structure	JPEG	JPEG 2000
4nn	Area	AF	23.4	25.0
		ASF	14.0	17.1
	Power	AF	16.2	25.2
		ASF	16.4 / 27.0	26.0
8nn	Area	AF	**12.8**	24.6
		ASF	14.2 / 23.6	**15.8**
	Power	AF	13.2	17.2
		ASF	13.6	25.2
Average for all results under 20:1			14.4	25.4

Table 7.21: The average compression ratio lying within the most frequently chosen range of Table 7.20.

The best two filters for JPEG and JPEG 2000 are highlighted in Table 7.21. Figure 7.72 shows the original Blackboard image, the filtered version, which uses the AF filter structure, 4nn connectivity and the area attribute with the visually lossless value of 12. The last image shows the filtered image compressed and decompressed using the JPEG CODEC to a compression ratio of 20:1 (0.4bpp), which is used as the reference image. The degraded images, the original image compressed and decompressed are shown in Figure 7.73. This shows the image with the lowest compression ratio, 1:1 (8bpp), that was voted for during testing, which shows no visible degradation at all. The second image shows the Blackboard image compressed to a ratio of 13:1, which is the average compression ratio voted for lying under a ratio of 20:1. Although some experts have said that they can see some degradation, few of the non-experts are able to see a difference. The final image shows the highest compression ratio, 32:1 (0.3bpp), which was voted for during testing. This shows significant degradation to the quality of the image, most especially in the background where blocks can easily be seen.

The JPEG 2000 results are shown in Figures 7.74 and 7.75. Figure 7.74 shows the original Blackboard image, a filtered version using the ASF filter structure, 4nn connectivity and the area attribute with the visually lossless value of 15. The last image shows the filtered image compressed and decompressed using the JPEG 2000 CODEC to a ratio of 20:1 (0.4bpp). The degraded images, shown in Figure 7.75, show that with the lowest compression ratio voted for, 12.2:1, and the average, 15.8:1, that there is very little visible degradation in the images. However, the highest amount of compression voted for, 35.3:1, shows distortions that can be seen by most of the observers. Many of the observers did comment that it was much harder to see differences with the JPEG 2000 CODEC than it was with the JPEG due to the sharp edges of blocks that are easy to see in JPEG.

a) Original 8bit greyscale, 720 x 576 Blackboard input image.

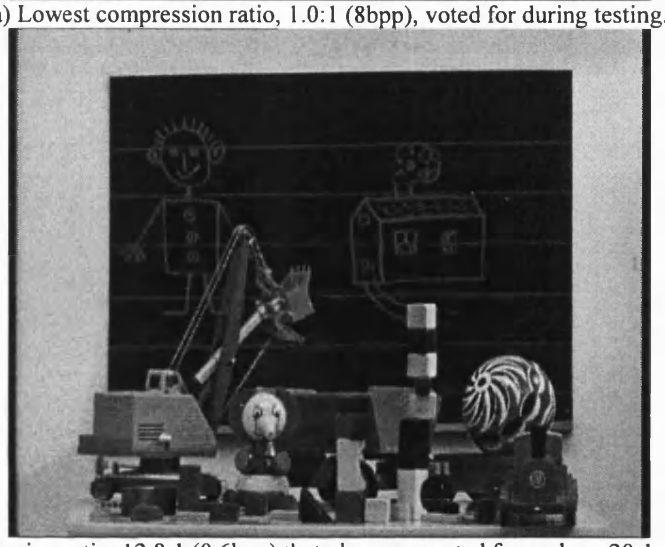


b) Image filtered using 4nn connectivity, the AF filter structure and the area attribute with the visually lossless value of 12.

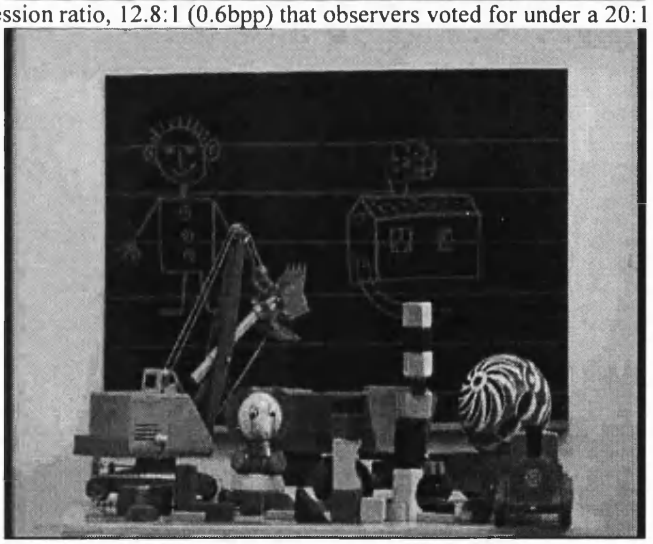
c) The filtered image (b) compressed and decompressed using the JPEG CODEC to a ratio of 20:1 (0.4bpp). This is used as a reference image in the visual test.

Figure 7.72: The input, filtered and reference image used in testing.

a) Lowest compression ratio, 1.0:1 (8bpp), voted for during testing.



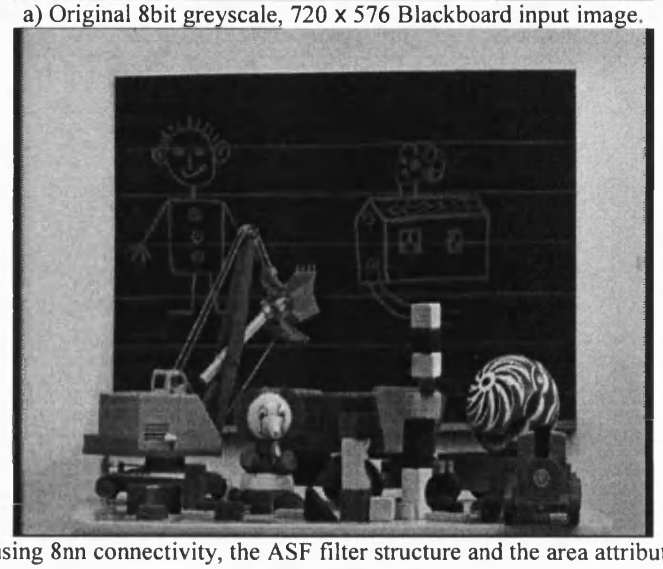
b) Average compression ratio, 12.8:1 (0.6bpp) that observers voted for under a 20:1 compression ratio.



c) Highest compression ratio, 32.1:1 (0.3bpp) that observers voted for.

Figure 7.73: The best JPEG result output, showing the lowest compression ratio voted for, the average of those lying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for.

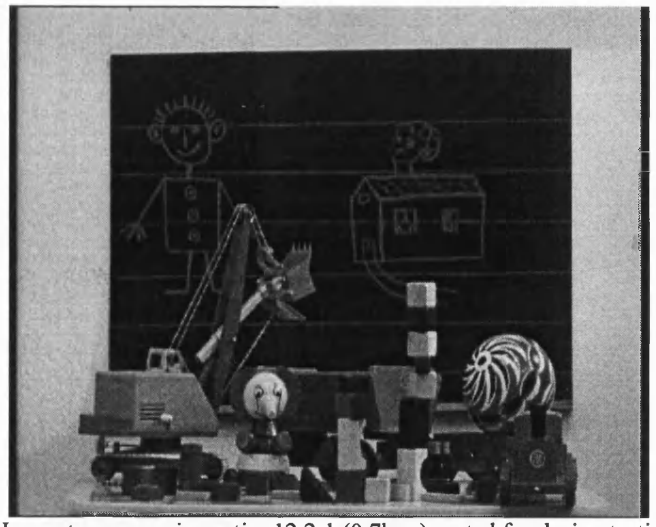
a) Original 8bit greyscale, 720 x 576 Blackboard input image.



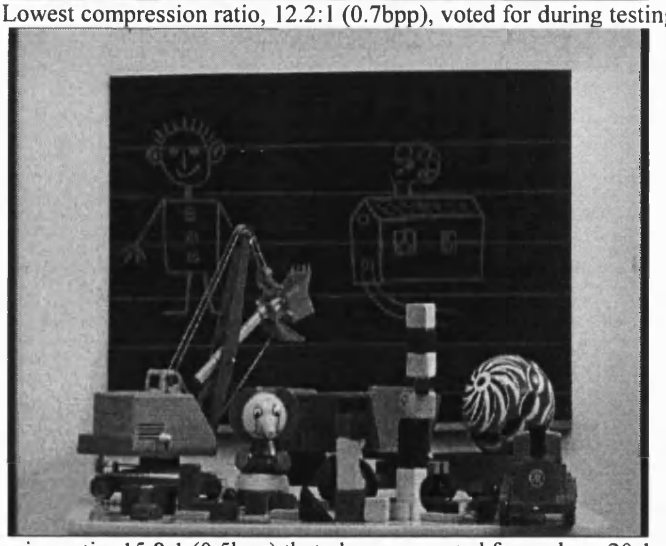
b) Image filtered using 8nn connectivity, the ASF filter structure and the area attribute with the visually lossless value of 15.

c) The filtered image (b) compressed and decompressed using the JPEG 2000 CODEC to a ratio of 20:1 (0.4bpp). This is used as a reference image in the visual test.

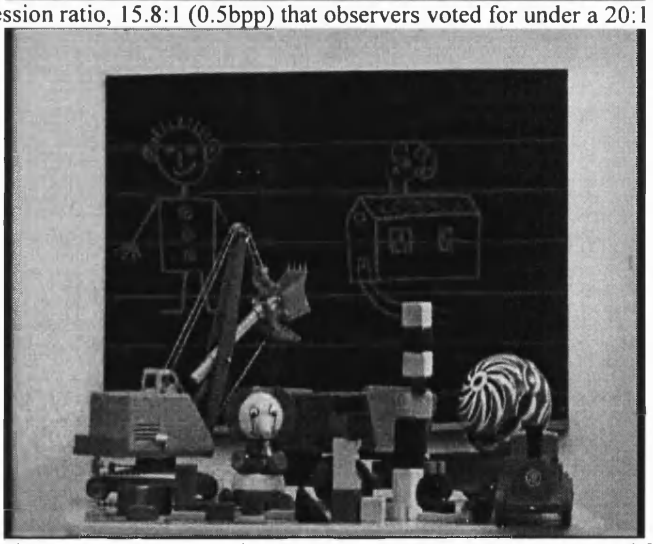
Figure 7.74: The input, filtered and reference image used in testing.



a) Lowest compression ratio, 12.2:1 (0.7bpp), voted for during testing.



b) Average compression ratio, 15.8:1 (0.5bpp) that observers voted for under a 20:1 compression ratio.



c) Highest compression ratio, 35.3:1 (0.2bpp) that observers voted for.

Figure 7.75: The best JPEG 2000 result output, showing the lowest compression ratio voted for, the average of those lying under a 20:1 (greater than 0.4bpp) compression ratio and the highest compression ratio voted for.

7.3 Conclusion

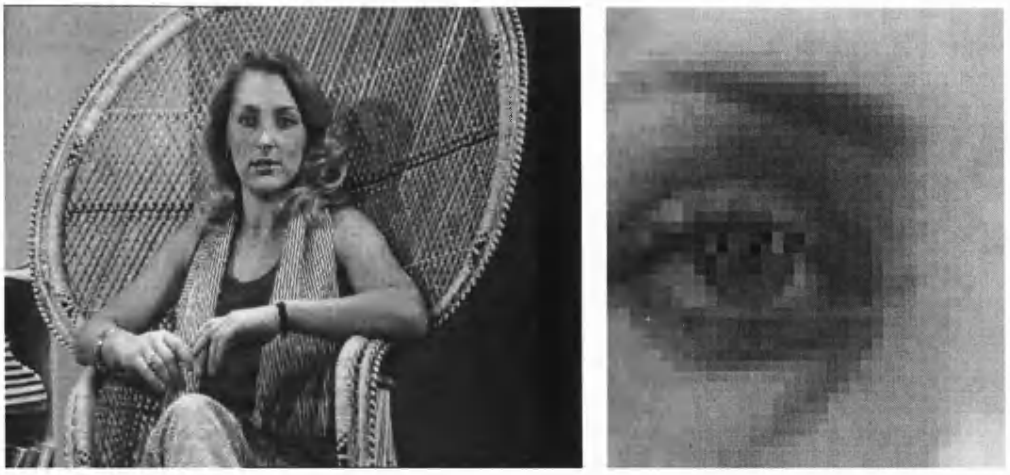
This chapter has shown two different ways in which mathematical morphology can be applied to preprocessing for image noise reduction and simplification. The first method described in section 7.1 showed how mathematical morphology could be used for image noise reduction. Both the AF and ASF filter structures (see section 4.6.2), 4nn and 8nn connectivity and area, contrast, volume and power attributes (see section 6.2.2) have been evaluated. This has shown, that the filter structure, connectivity and attribute used depend largely on the amount of noise present in the image. With a high degree of noise, 14dB the area attribute performs better than any other. The contrast attribute produces better results than the area only when there is little noise. Neither of these attributes are consistent though. For example, the contrast attribute performs best using 4nn connectivity and the ASF filter structure on the Barbara image corrupted with 14dB of noise. However using less noise, 35dB and the AF filter structure and 4nn connectivity produce the optimum results. The volume attribute produces the optimum result using the 4nn connectivity and the ASF filter structure, regardless of how much noise is present. In addition, the volume attribute shows clearly that the 8nn connectivity results do differ depending on the filter structure used. However, with the exception of a high degree of noise, the power attribute performs significantly better than all other attributes. For most of the test images (Barbara, Barbara 2, Boats and Goldhill), except for images with 14dB of noise, the AF filter structure and 4nn connectivity obtain the optimum output. Thus although the power attribute is more computationally intense, the complexity is reduced by the use of the AF filter structure. Most of the other attributes show the optimum when the ASF filter structure is used with 4nn connectivity. Thus although the area, contrast and volume attributes are less complex, the ASF filter structure is far less efficient than the AF. It is also apparent that as the noise increases, so does the attribute value. This is due to the fact that more noise must be removed, which directly corresponds to requiring a larger attribute. If the amount of noise affecting an image is known, or a close guess is available, then the filtering can be accomplished by using a measure, the mean for example, of the optimum values found for the test images at that level.

However, for the noise reduction, the amount of noise present is not always known. The amount of noise could be estimated using blind estimation **[143]**. A similar approach to estimate the noise variance has been used to apply both noise filtering (see section 7.1.3) to unseen data, thus proving that the morphological filtering techniques can be successfully applied to any data and still improve the compressibility. It has also been shown that mathematical morphology can be used to process images whilst not degrading the visual quality (see section 7.2). This is useful for removing visually redundant information that would otherwise use bandwidth unnecessarily. The results from this section show that higher attribute values are acceptable when using 8nn connectivity. It was also shown that two images that were not used to train the system also can be filtered using this method without any visual degradation. It would be beneficial to use more images to train all of the systems developed, allowing for more accurate estimation of the attribute parameters for noise reduction and allow for a more suitable visually lossless value to be found. In addition, a much larger observer base would be beneficial to increase the feedback and eliminate any spurious data.

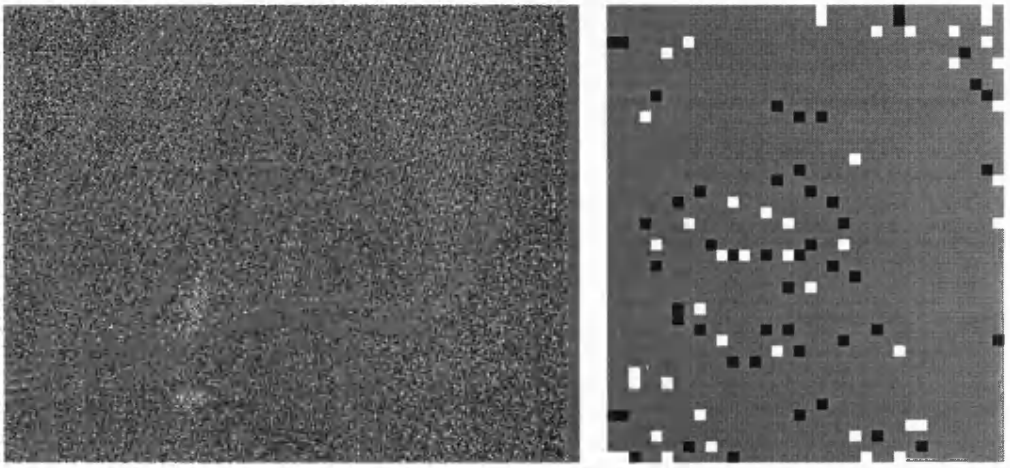
8 Intermediate Area-Morphology

Although the morphological filtering methods shown in chapter 7 have been shown to be effective, they only processes the minimum and maximum parts of the image. Some images will contain a large number of these regions whilst others may contain very few. In addition, in many cases, parts of the image are fairly flat but are neither a minimum nor maximum. This can be overcome to a certain extent by changing the value of h (see section 6.1.1.1). Another solution is to use Intermediate Regions. An intermediate region can be defined as a Flat Zone [73], [144] in the image that does not constitute a minimum or a maximum region. A flat zone is a region of pixels, which have exactly the same value (this includes minimum and maximum regions). However, as described earlier, noise can cause fluctuations in the image. Thus it is required that this is taken into account. Hence, an intermediate region can be redefined as a region of pixels, which have the same value within a given tolerance and are not contained in a minimum or maximum region. This is shown more clearly in Figure 8.1. The top image shows the input image whilst the middle and bottom show the seed regions and the regions at size 100. As can be seen, the majority of the image is not contained in a region (the grey areas) or are removed in the early stages of the growing algorithm.

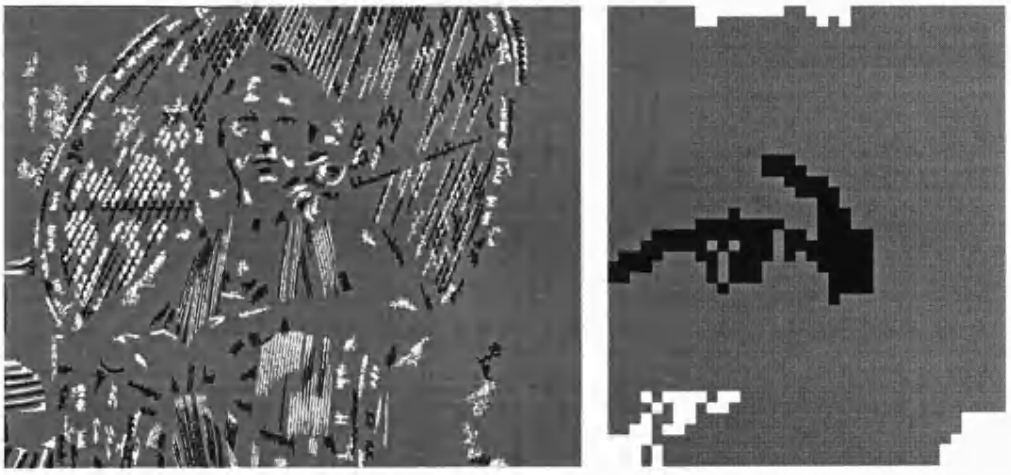
There are two possible methods for implementing intermediate morphology. One is to combine the intermediates with the minima and maxima in an AOC (see section 4.6.2). This is termed an Intermediate AOC. The second method is to extract all flat zones and then grow them all as intermediates known as Full Intermediate AM. For both methods, the value of the region is the mean value of the pixels contained within it. Also, an intermediate region, unlike a minimum or maximum, grows to the pixel or region that has the closest value. However, this causes problems with the priority queue method as the neighbours no longer have the same priority (distance) on each pass and so have to be dynamically reordered. This could be done by re-evaluating the entire priority queue on each pass, but this decreases the efficiency of the code. Hence, the priority queue is considered useless for the intermediate regions. Since all of the neighbours need to be re-evaluated on each pass anyway, a more efficient method is to store the neighbours as an array in the same way as the pixels in the region are stored. This shows that an intermediate region will be far less efficient, in terms of computational power than the minima or maximas.



a) Input image – Barbara 2, left eye enlarged on the right



b) Seed regions. White maxima regions, black minima regions

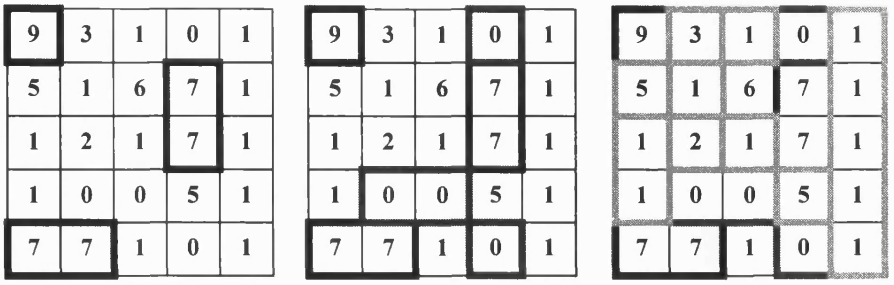


c) Region after growing to a size of 100.

Figure 8.1: Example to show how much of the image (Barbara 2 at 8bit greyscale, 720 x 576 pixels) is actually affected.

8.1.1 Implementing Intermediate Area-Open-Close

Firstly, as in any other method the regions are extracted. This is done by first extracting the minima and maxima regions. From the areas that are not in a region, the flat zones are extracted (see Figure 8.2). The regions are then grown in the same way as before. That is, each region of size 1 is grown, then size 2. This process repeats until a set target is reached.



Extract the local and regional maximums.

Extract the local and regional minimums.

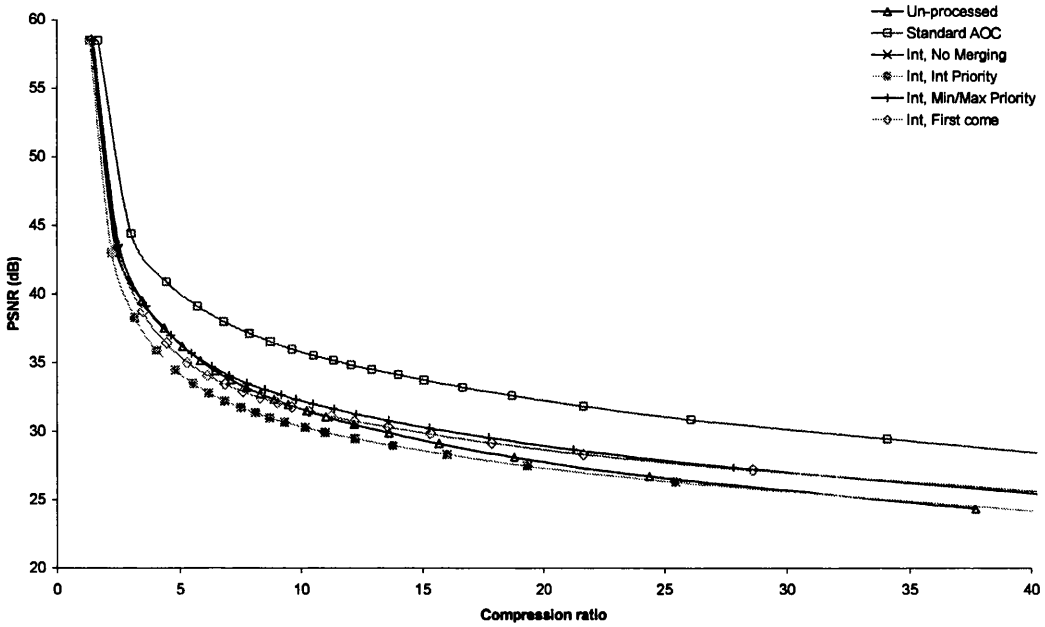
Extract the local and regional flat zones.

Figure 8.2:Example of how regions are extracted and labelled.

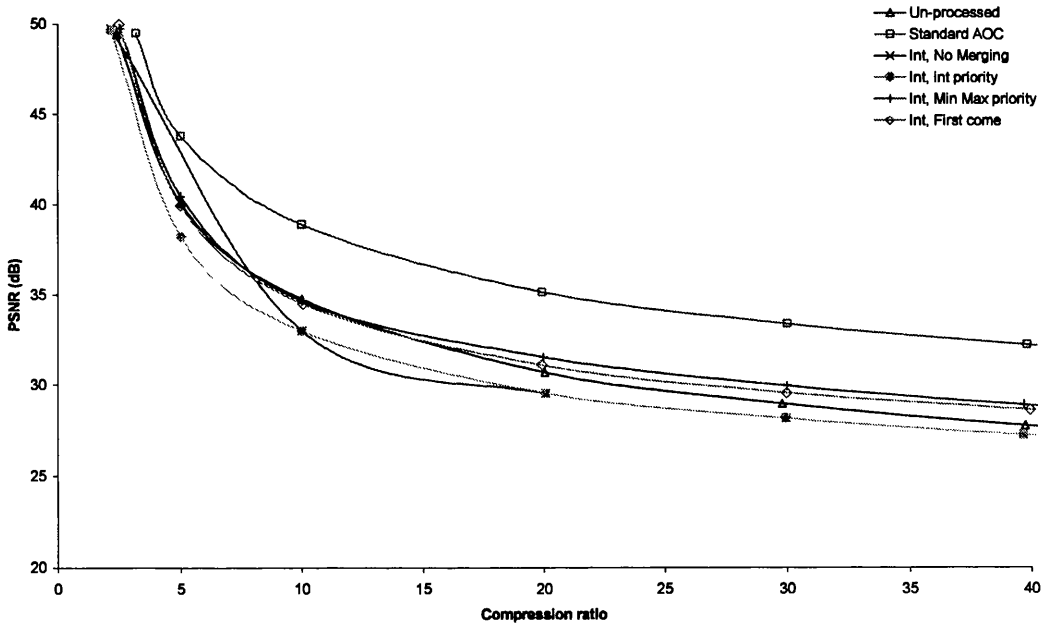
However, there is now the problem of what happens when a region collides with an intermediate region. There are four possible actions to take:

- Do not allow any regions to merge
- Allow intermediates to have priority, hat is any region trying to merge with an intermediate will become an intermediate.
- Allow minima and maxima to have priority, that is any minima or maxima region merging with an intermediate, will cause the intermediate to become a minima or maxima.
- Allow the first come to have priority.

Results of these methods are shown in Figures 8.3 – 8.5. As can be seen from the results, the best method is to allow minimum and maximum regions to have priority. Unless stated otherwise, this is the method that will be used from this point forward. However, these results also show that this method actually degrades the CODEC performance in places, making it harder to compress than the unprocessed image. Also, none of the intermediates comes close to the result obtained from a standard Area-Open-Close of the same size.



a) Rate distortion for the Barbara 2 image using intermediate morphology using the JPEG CODEC.



b) Rate distortion for the Barbara 2 image using intermediate morphology using the JPEG 2000 CODEC.

Figure 8.3: PSNR (dB) against compression ratio using the intermediate methods.

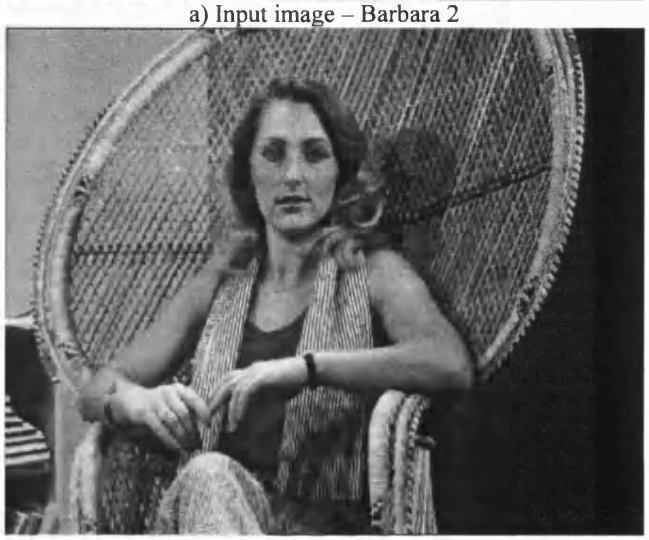
a) Input image – Barbara 2

b) First come priority

c) Intermediate regions have priority

Figure 8.4: Example of intermediate filtered images using the 8bit greyscale 720 x 576 Barbara 2 image.

a) Input image – Barbara 2



b) Minima and maxima regions have priority.

c) No merging is allowed between different region types.

Figure 8.5: Example of intermediate filtered images using the 8bit greyscale, 720 x 576 Barbara 2 image.

An intermediate region will grow until the target area size is reached. It is suspected that this is what is causing the degradation in performance. This can change region values by a large amount. Therefore, to improve performance a mechanism is needed to reduce the amount of processing done on the image by intermediate morphology. Two methods have been used to attempt this:

- Use a morphological gradient to select intermediate seed regions,
- Use a closest match function to select intermediate seed regions.

Both methods are looked at in the following two sections.

8.1.2 Intermediate Morphological Gradient

Instead of using all flat zones as intermediate regions, the Morphological Gradient is used as a mask. First the image is processed with the morphological gradient as given by:

$$G(A,B)=\frac{(A\oplus B)-(A\ominus B)}{2} \tag{8.1}$$

where A is the image and B is the structuring element. Morphological reconstruction is then used to extract the minimums from this in the same way that the minimum and maximum regions were extracted. The regions are then labelled as intermediates (see Figures 8.6 and 8.7). If a pixel is already in a minimum or maximum region, then they are left in that region. This method is then evaluated in the same way as before. The results are shown in Figures 8.8 and 8.9. The results show this to be an improvement upon the previous method, however it still does not perform as well as the standard Area-Open-Close method.

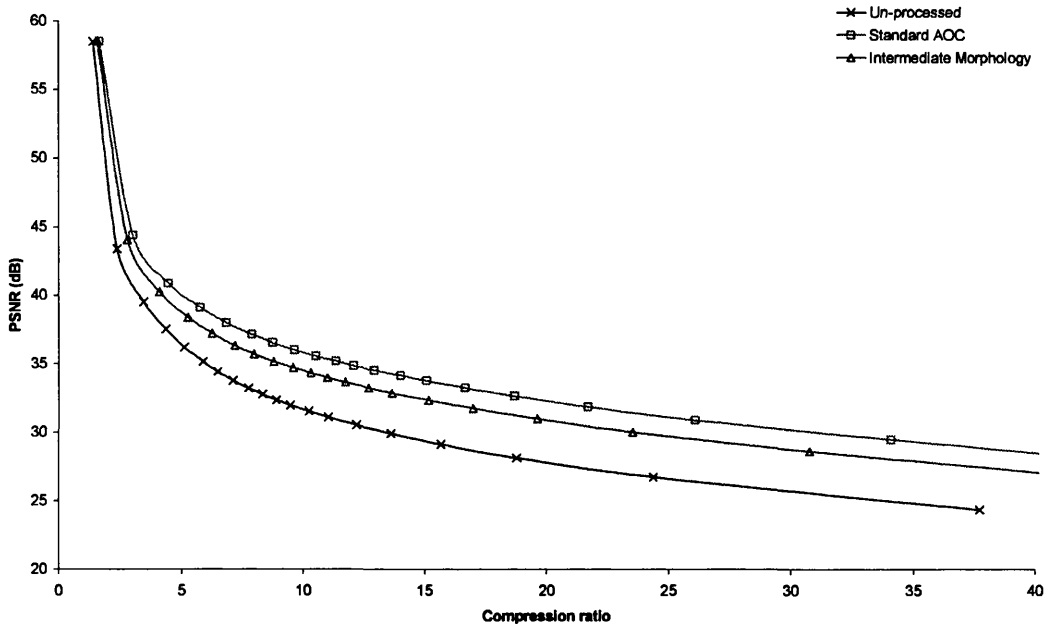
Figure 8.6: The 8bit greyscale, 720 x 576 Barbara 2 image used as the input for intermediate morphology filters shown in Figure 8.7.

a) Gradient of the image (note it has been scaled to make the image visible).

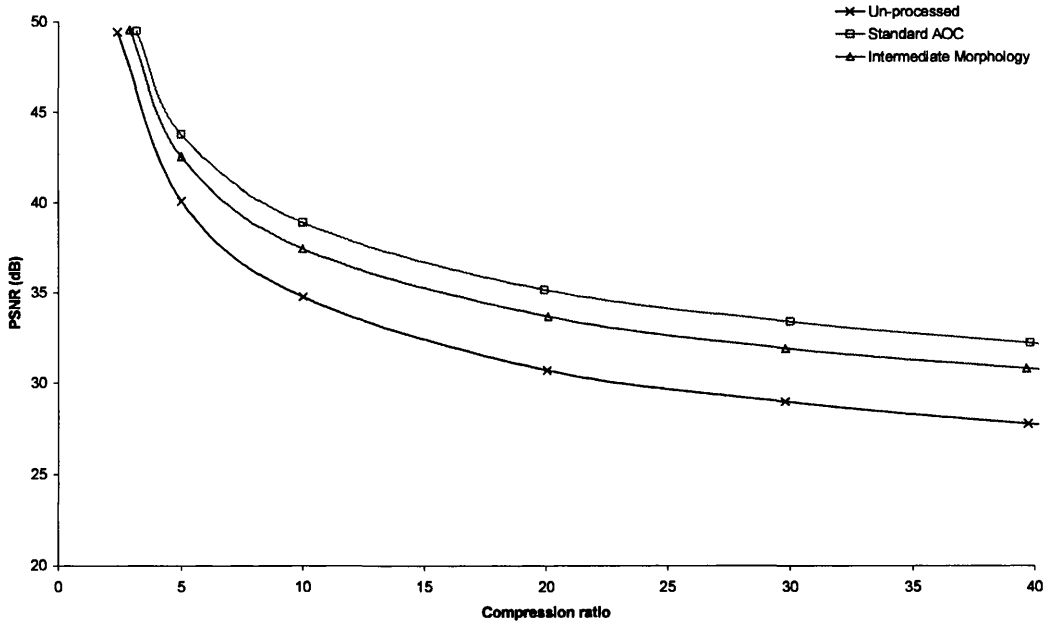
b) Reconstructed gradient (Again this has been scaled).

c) Extracted intermediate regions.

Figure 8.7: Region extraction using the morphological gradient.



a) Rate distortion of the Barbara 2 image after filtering with the intermediate morphology using the JPEG CODEC.



a) Rate distortion of the Barbara 2 image after filtering with the intermediate morphology using the JPEG 2000 CODEC.

Figure 8.8: Intermediate morphological results for an compressing the Barbara 2 image after filtering using an area size of 100.

a) Input image – Barbara 2.

b) Intermediate filtered image using the gradient method with an area size of 100.

Figure 8.9: Example to show the result of the gradient intermediate method using the 8bit greyscale, 720 x 576 Barbara 2 image.

8.1.3 Intermediate Distance

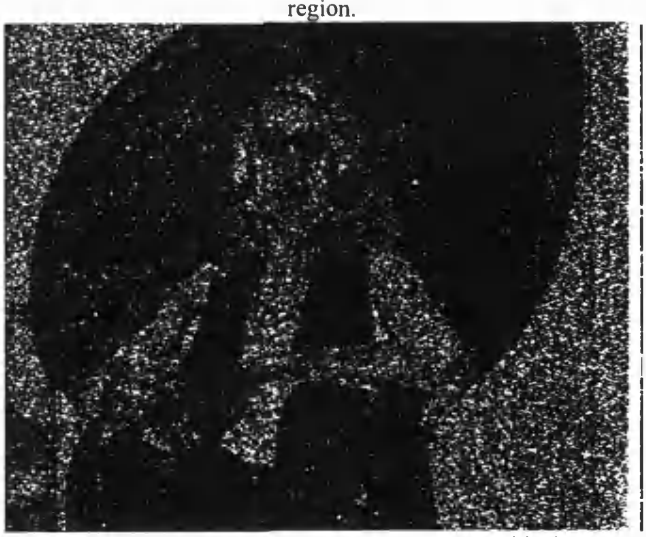
The second method that has been used to try to improve intermediate morphology is to use a distance function as given by:

$$D_{x,y}(B) = \min_{u,v \in B}\left(\left|A_{x,y} - A_{x+u,y+v}\right|\right) \tag{8.2}$$

This finds the closest value to the current pixel. If a neighbouring pixel is the same value, then the distance will be 0. The transform is applied on the entire image with the resulting image will mark any flat regions with a 0. Any area set to 0 is then extracted. This is done such that all the pixels with the same value are grouped together. Figure 8.10 shows this more clearly. The results for this method are shown in Figure 8.11 and Figure 8.12. Unfortunately the performance is not improved much.

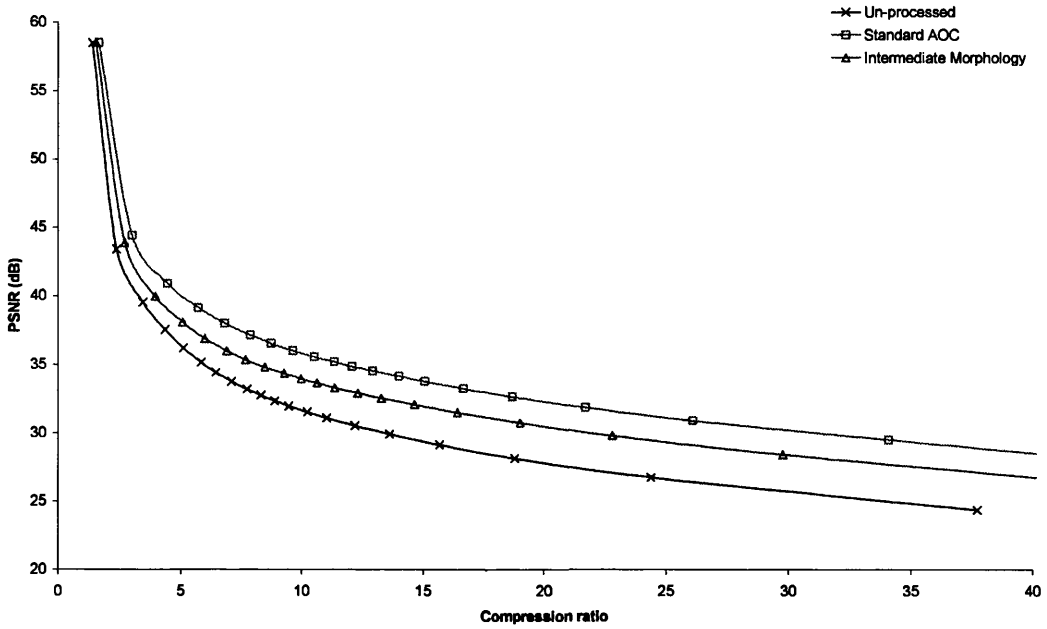
a) Input image – Barbara 2.

b) Result from the distance function. This has been scale to make it visible. Black indicates a flat region.

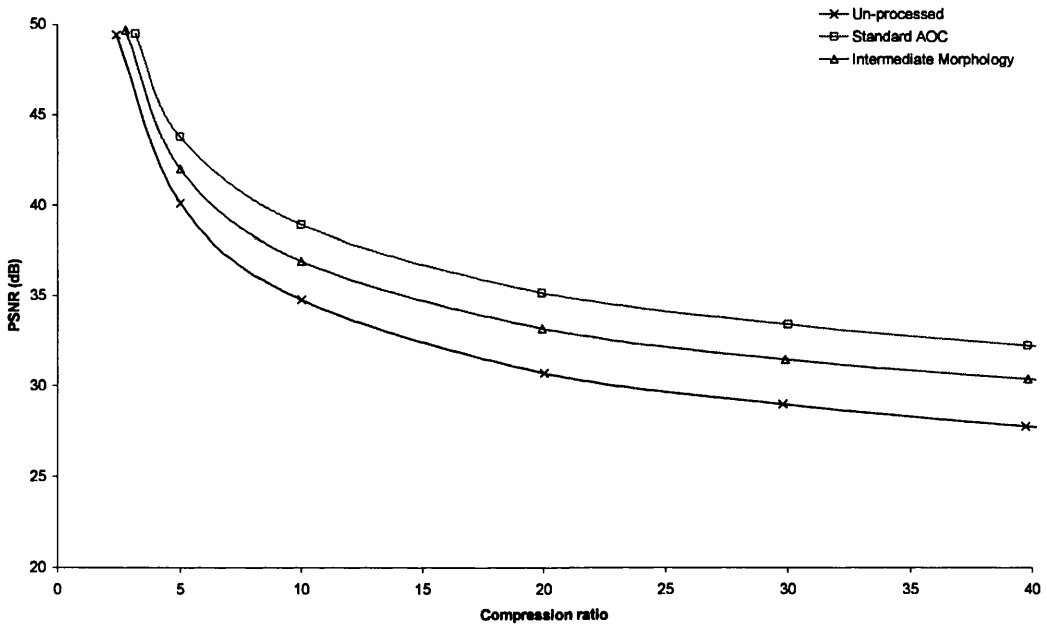


c) Extracted intermediate regions (shown as black).

Figure 8.10: Sample using the 8bit greyscale, 720 x 576 Barbara 2 image to show the regions extracted by the distance method.



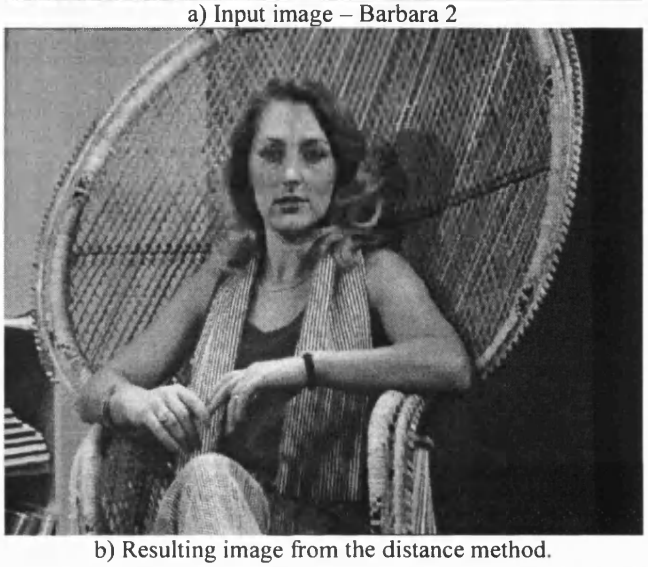
a) Rate distortion graph for the Barbara 2 image using the JPEG CODEC.



b) Rate distortion graph for the Barbara 2 image using the JPEG 2000 CODEC.

Figure 8.11: Performance of the intermediate morphology distance method.

a) Input image – Barbara 2

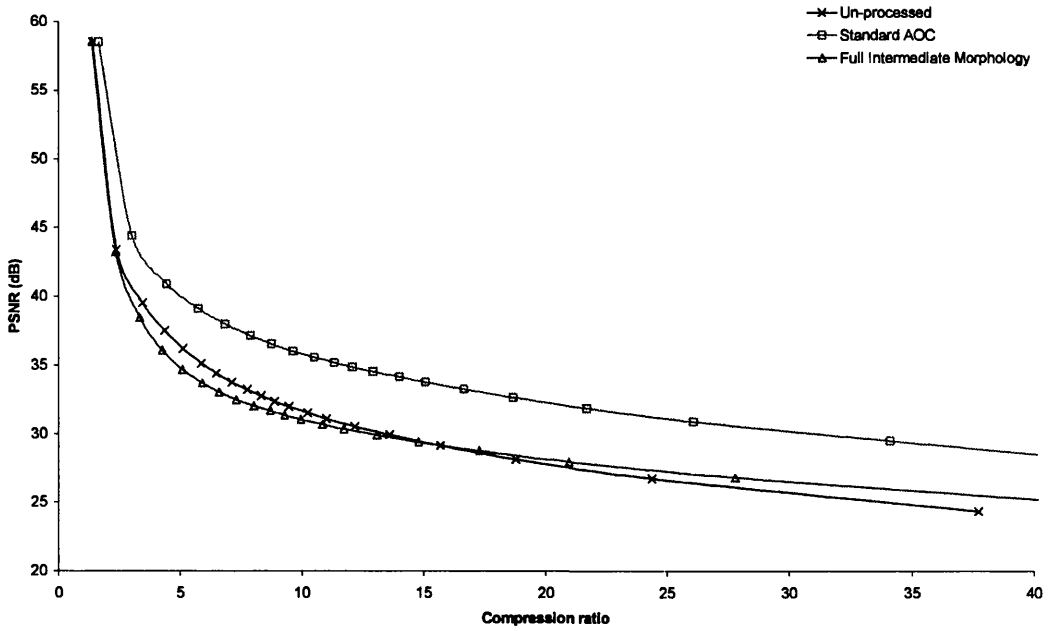


b) Resulting image from the distance method.

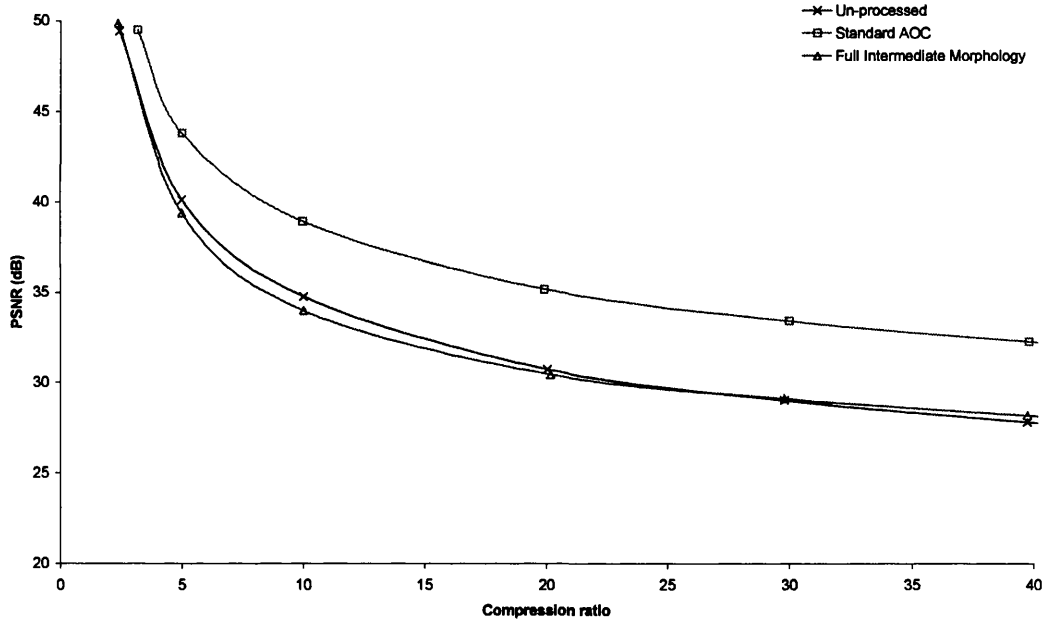
Figure 8.12: Sample images to show the result of the gradient intermediate method.

8.1.4 Full Intermediate Area Morphology

This is the second method that has been implemented for attempting an implementation of intermediate area morphology. In this method, the flat zones for the entire image are extracted so that every pixel is in a region. The regions are then grown as was done in the previous methods. This method is more complex than any of the others attempted. However, some advantages are gained through this method. For example, in previous methods, if a neighbouring pixel were already in a region, then the list of regions would have to be traversed in order to find the region. In some cases be very time consuming and inefficient. Since every pixel in this method is in a region, the neighbours are stored as pointers to the region that the neighbour is in. Hence the neighbouring region is available instantly. Figures 8.13 and 8.14 show results of this method. As can be seen, there is not much of an improvement in performance and is perhaps the worst of all the methods attempted.



a) Rate distortion graph using the full intermediate area morphology and the JPEG CODEC.



b) Rate distortion graph using the full intermediate area morphology and the JPEG 2000 CODEC.

Figure 8.13: PSNR (dB) against compression ratio to show the performance of the full intermediate method.

a) Input image – Barbara 2

b) Resulting image from using the full intermediate method to an area size of 100.

Figure 8.14: Sample images to show the result of the full intermediate method on the 8bit greyscale, 720 x 576 Barbara 2 image.

8.2 Conclusion

This chapter has attempted to widen the morphological approach so that all pixels in the image would be affected. Whilst some morphological operators do this, the standard dilation for example, it was hoped that this technique would provide more control over the filtering process and give a better performance. However, whilst some of these intermediate methods do still improve the compression performance, none give as good a performance as the standard Area-Open-Close methods described in chapter 7. Hence it has been concluded that there is little to be gained from the intermediate methods and hence no further development has been attempted.

9 3D Preprocessor Development For Noise Reduction

Chapters 6 and 7 developed a spatial preprocessing systems for digital images. This chapter looks at developing this further to form a spatio-temporal (3D) preprocessing system for digital video. This is advantageous in video processing, as adjacent frames are in most cases correlated, whereas noise is uncorrelated, and therefore the frames contain redundancy. In addition, because motion is involved, the properties of the HVS (see chapter 3) can be used to increase the processing of the video. For example, objects that appear only on one frame can be removed, as the previous and next frames will mask this object. Since the 2D methods showed that the power attribute removed more noise than any other attribute, this is the main attribute used for the spatio-temporal filtering, with the area attribute used to provide a comparison. Two sequences, Kiel and Foreman, are used for evaluation after being corrupted with AWGN (see section 5.1.1) to give a PSNR of 28.12dB and 28.14dB respectively. The sequences are Common Image Format (CIF) sized, 352 x 288 at 25fps, with 50 frames for the Kiel sequence and the first 150 frames of the Foreman sequence being used. Only the luminance (y) channel is used making them 8bit greyscale. Hence the uncompressed bandwidth is 19.34Mbps. All of the CODEC evaluations are performed at a compression ratio of 20:1, which gives 0.4bpp or 990Kbps.

All of the methods developed in this chapter use the pixel queue algorithm (see section 6.1) and the Efficient ASF method (see section 6.3.1). Although this is one of the slowest methods to execute, it is used due to the fact that it is much easier to experiment with and measure various parameters with than both the Max-tree (see section 6.1.4) and Wilkinson's (see section 6.1.5) methods. Hence this chapter is concerned with the development of new spatio-temporal filtering methods and not the efficiency of the implementation. Thus several of the algorithms described are very slow but could be optimised in future work.

9.1 Spatial Processing of Video

Before proceeding with the development of 3D morphological filtering methods, two current methods are used as a comparison. The first technique is the standard spatial Gaussian filter (see section 5.3.1.2) and the second is the spatio-temporal K-NN approach (see section 5.3.3), which will then give a basis for the morphological methods to out perform. Two sequences, Foreman and Kiel are used, after being corrupted with AWGN (see section 5.1.1). These are then filtered, compressed and decompressed using the four CODEC's listed below, which provides an additional basis for improvements to be made upon:

- MJPEG,
- MPEG-II [18],
- H263 [25],
- BWV [142],

9.1.1 Gaussian Filtering

The Gaussian filtering method (see section 5.3.1.2) is used for a comparison, as the Gaussian is a well known and understood filtering tool. Although it is possible to extend the Gaussian function into other dimensions, a simple spatial only method is used. Thus each frame is filtered independently, which avoids the need for any complicated motion tracking system to allow for spatio-temporal filtering. The two sequences are corrupted with AWGN (see section 5.1.1) so that the noisy images, when compared to the originals, have a PSNR (see section 3.2.1) of 28.14dB and 28.12dB for the Foreman and Kiel sequences respectively. Gaussian filtering is then applied to the corrupted sequences. However, as with the noise removal for images (see section 7.1), there are two variables, σ and the mask size that need to be selected, which is accomplished by iterating through different values for each. The value of σ is varied from 0.1 to 1.0, in steps of 0.1 and the mask size is varied from 3 x 3 to 27 x 27 in steps of 2. The results shown in Figure 9.1 show that like the spatial results, the mask size and value of σ appear to depend upon the amount of noise present in the image.

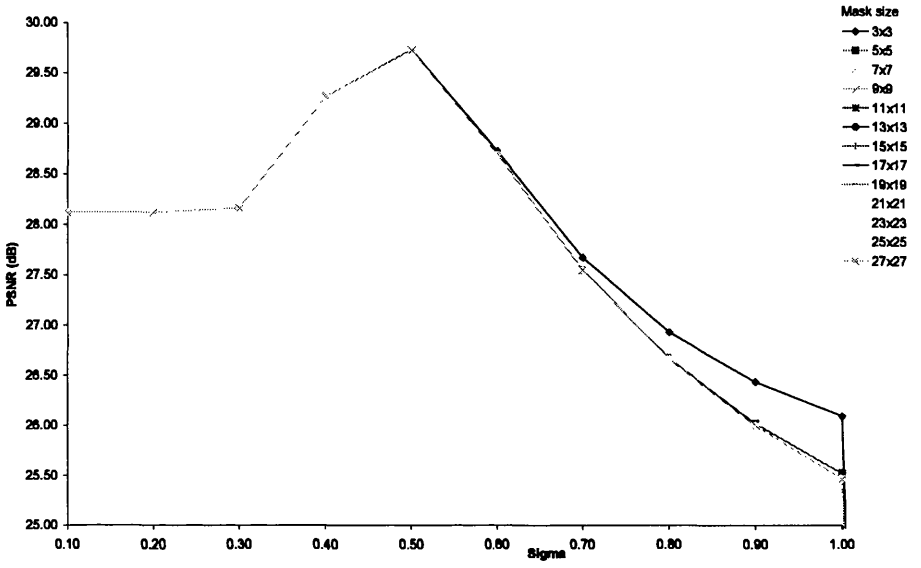
The best filter outputs are shown in Table 9.1. The Kiel sequence (see Figure 9.1a) produces the best filtered output using a σ of 0.5 and a mask size of 3 x 3 giving a PSNR of 29.74dB when compared to the uncorrupted original, thus giving an increase in performance of 1.62dB (i.e. 1.62dB of noise is removed from the image. Results for the Foreman sequence (see Figure 9.1b) show that the best filter uses a 3 x 3 mask size as did the best filter for the Kiel sequence but uses a σ of 0.6 to give a PSNR of 32.11dB. This results in a much larger performance, 3.97dB, than the Kiel sequence showed. This is due to the fact that the foreman sequence has less detail, or edges, than the Kiel sequence, and therefore less critical information is removed through filtering.

Sequence (Noise level (dB))	σ	Mask size	PSNR (dB)
Kiel (28.12dB)	0.5	3 x 3	29.74
Foreman (28.14dB)	0.6	3 x 3	32.11

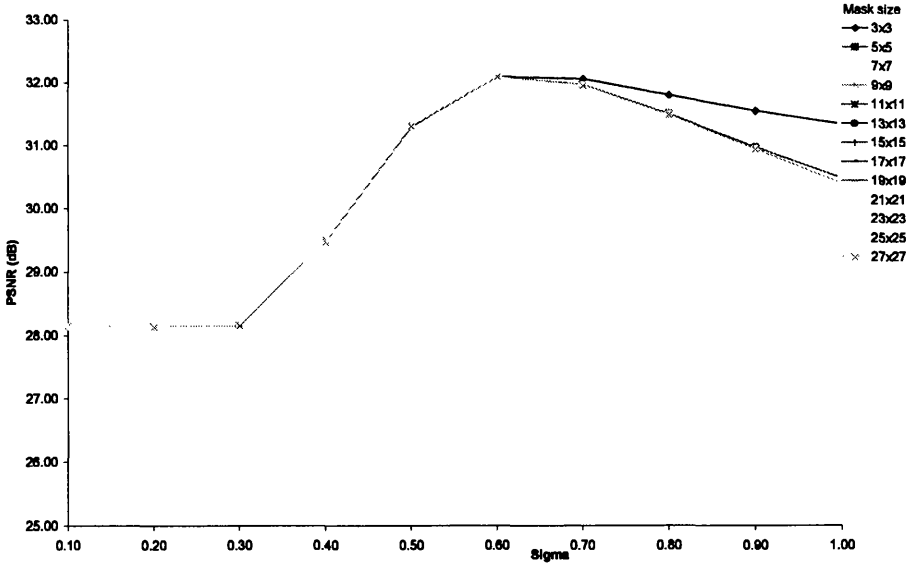
Table 9.1: Optimum filter outputs using the Gaussian filtering method.

Figures 9.2 and 9.4 show the original and corrupted frames 10 and 100 respectively of the Foreman sequence whilst Figures 9.3 and 9.5 show the filtered outputs for frames 10 and 100 respectively. The first images of each, Figures 9.3a and 9.5a, show the best filter output. The second image, Figures 9.3b and 9.5b, show results using a mask size of 9 x 9 and a σ of 1.0. This lies between the best and worst outputs and gives a PSNR of 30.40dB, which is still an improvement although blurring of the frames is apparent. The last image, Figures 9.3c and 9.5c, show the worst filter output, which used a

mask size of 3 x 3, although any mask size up to and including 27 x 27 produced the same result, and a σ of 0.1 and gave a PSNR of 28.14dB. Hence the worst output is the same as the noisy input sequence. This has shown that the Gaussian filtering method can be applied spatially to video sequence and successfully reduce the noise present in the sequence.



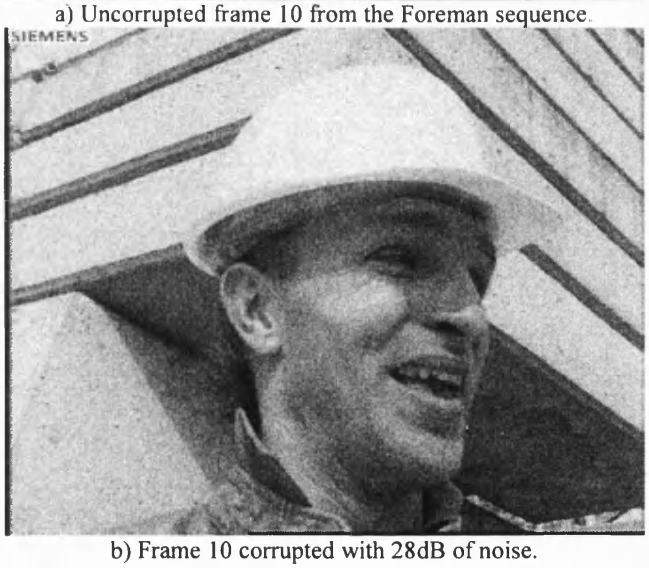
a) Filter output for the Kiel sequence.



b) Filter output for the Foreman sequence.

Figure 9.1: Gaussian filter performance.

a) Uncorrupted frame 10 from the Foreman sequence.

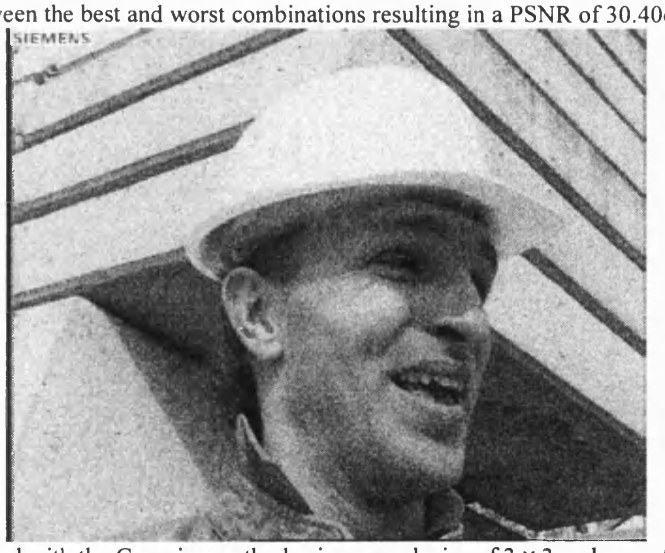


b) Frame 10 corrupted with 28dB of noise.

Figure 9.2: Original and corrupted of frame 10 of the 8bit greyscale, 353 x 288 Foreman sequence.

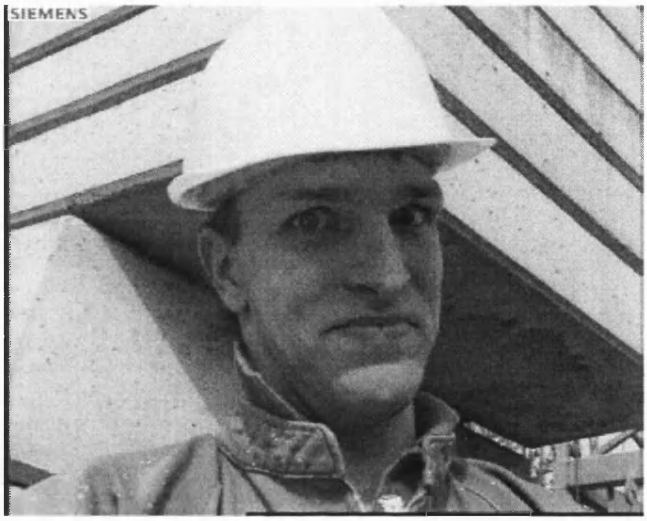
a) Frame 10 filtered with the Gaussian method using a mask size of 3 x 3 and a σ of 0.6. This is the best filter output resulting in a PSNR of 32.11dB.

b) Frame 10 filtered with the Gaussian method using a mask size of 9 x 9 and a σ of 1.0. This lies between the best and worst combinations resulting in a PSNR of 30.40dB.

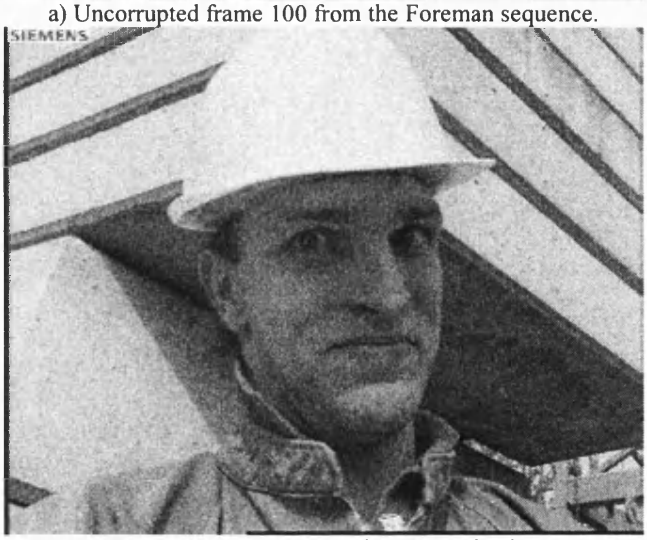


c) Frame 10 filtered with the Gaussian method using a mask size of 3 x 3 and a σ of 0.1. This is the worst filter output resulting in a PSNR of 28.14dB.

Figure 9.3: Gaussian filtering of frame 10 of the Foreman sequence.

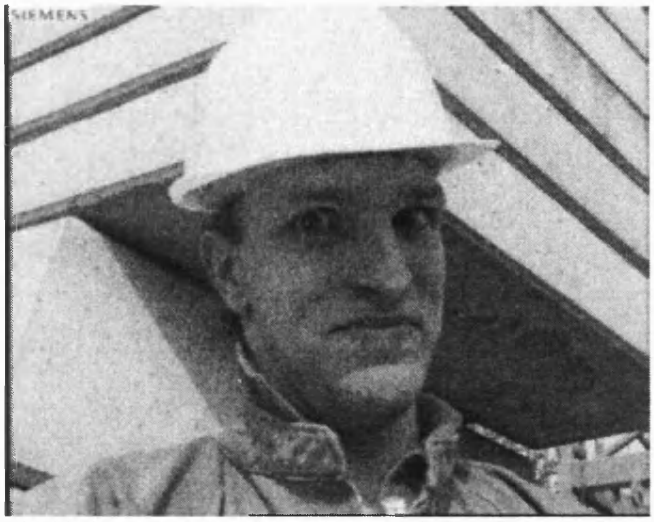


a) Uncorrupted frame 100 from the Foreman sequence.

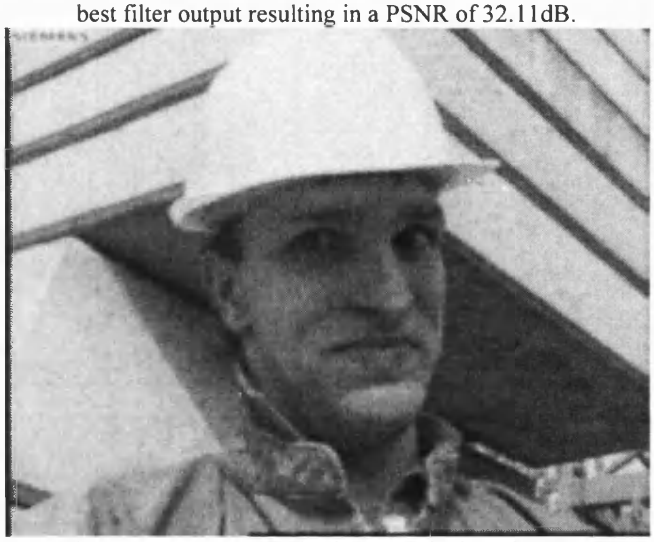


b) Frame 100 corrupted with 28dB of noise.

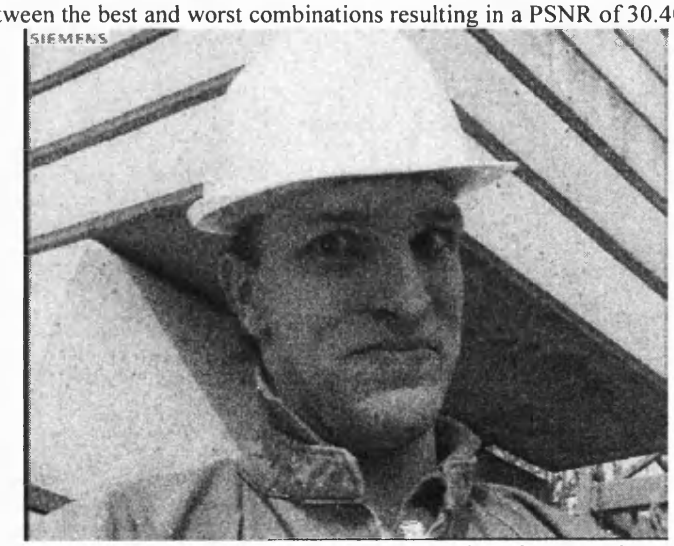
Figure 9.4: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.



a) Frame 100 filtered with the Gaussian method using a mask size of 3 x 3 and a σ of 0.6. This is the best filter output resulting in a PSNR of 32.11dB.



b) Frame 100 filtered with the Gaussian method using a mask size of 9 x 9 and a σ of 1.0. This lies between the best and worst combinations resulting in a PSNR of 30.40dB.



c) Frame 100 filtered with the Gaussian method using a mask size of 3 x 3 and a σ of 0.1. This is the worst filter output resulting in a PSNR of 28.14dB.

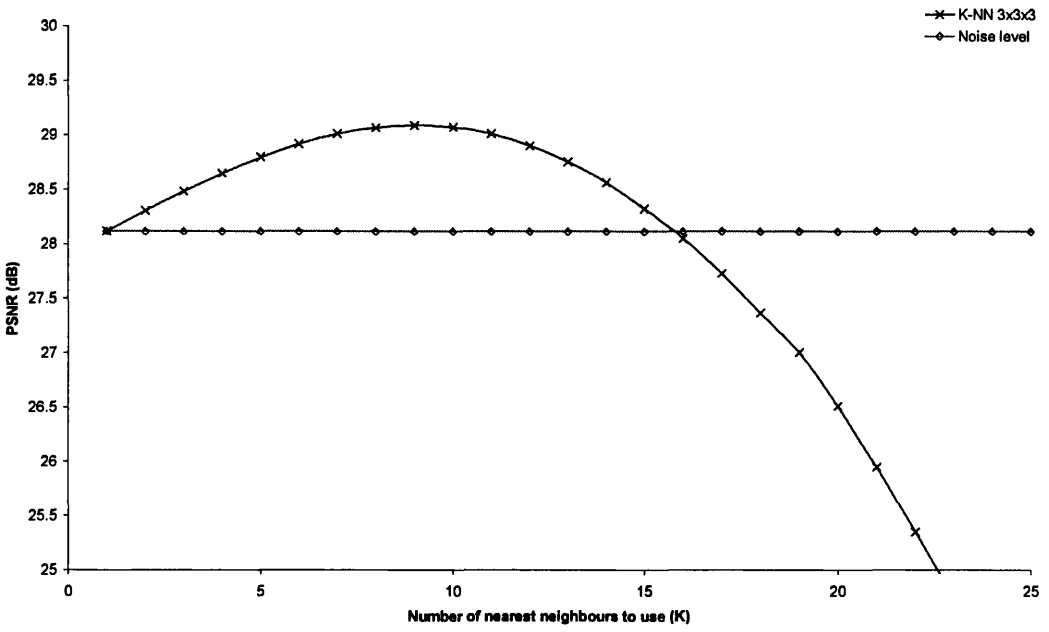
Figure 9.5: Gaussian filtering of frame 100 of the Foreman sequence.

9.1.2 K-Nearest Neighbour Filtering

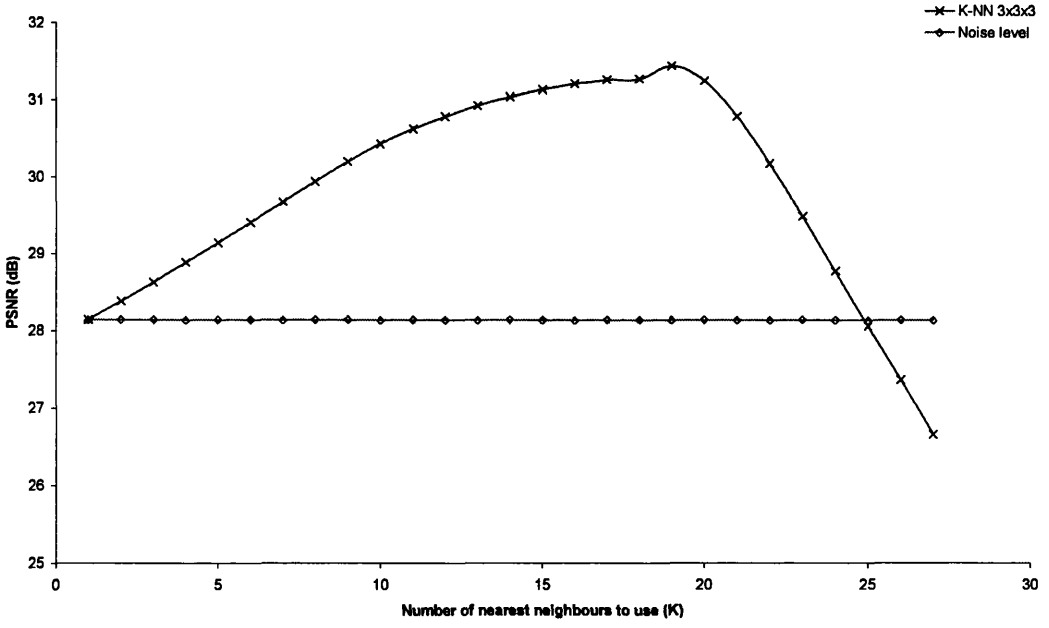
The K-Nearest Neighbour (K-NN) approach (see section 5.3.3), is used as a second comparison. This is a very simple filtering system that has been shown to work very effectively. This works by filtering a frame using the two adjacent frames to calculate the output for the current frame. A window around the current frame is extracted and the pixels ranked according to their distance, in intensity, from the current pixel. The output of the filter is then the average of the K closest values. A block window of 3 x 3 x 3 pixels is used, thus only the number of neighbouring pixels to be used is to be found. This is found by iterating through various values of K, that is the number of nearest neighbours to the current pixel to use, starting from 2 and finishing at 27. Note that a value of 1 will not actually filter the image and a value of 27 converts this filter to a mean filter.

Figure 9.6 shows the results for the K-NN filter. This shows that best amount of nearest neighbours to use differs for both sequences with the optimum filtering for the Kiel sequence being found to be when 9 of the nearest neighbours are used giving a PSNR of 29.10dB and 19 of the nearest neighbours used for filtering the Foreman sequence to achieve the optimum PSNR of 31.44dB. Both sequences show that too much filtering, for example when all of the neighbours are used, the output of the filter has a PSNR value that is worse than the noisy sequence. For example if all of the neighbours are used on the Kiel sequence, the filter is actually then just a mean or moving average, the resultant PSNR is 22.35dB. Figures 9.7 and 9.9 show the input frames 10 and 100 respectively from the Foreman sequence both uncorrupted and corrupted with 28dB of AWGN (see section 5.1.1). Figures 9.8 and 9.10 show the filtered frames with the first image (Figures 9.8a and 9.10a) showing the best filter output, which is produced by using the 19 nearest neighbours to give a PSNR of 31.44dB. The second image, Figures 9.8b and 9.10b, show a result using 5 of the nearest neighbours, which is in-between, the best and worst results giving a PSNR of 29.15dB. The last image, Figures 9.8c and 9.10c, shows the worst filter combination. This used all of the neighbours, which is the same as a moving average or mean filter, and resulted in a PSNR of 26.66dB. Thus the worst filter actually increases the degradation to the sequence by 1.48dB.

Visually the best filter (see Figures 9.8a and 9.10a) shows clearly that noise is reduced as does the results that lay in-between the best and worst results (see Figures 9.8b and 9.10b), although noise is still visible. The worst result (see Figures 9.8c and 9.10c) has actually removed a great deal of noise, but in doing so has also removed a significant amount of detail, which has resulted in a blurred sequence. Compared to the Gaussian method (see section 9.1.1), the results of this method are not as good, but are not too far away. For example the Kiel sequence has a best PSNR of 29.74dB for the Gaussian filter and 29.10dB for the best K-NN filter. However, the K-NN method has only one parameter that needs to be found and requires no mask to be computed or stored, thus resulting in a more efficient algorithm.

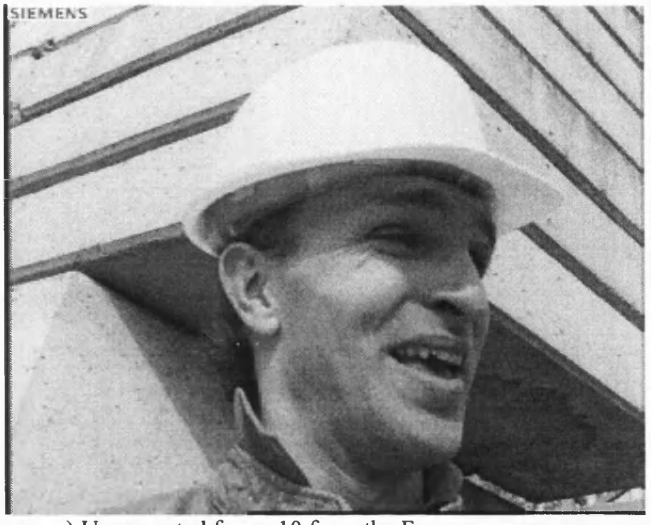


a) Filter output for the Kiel sequence.

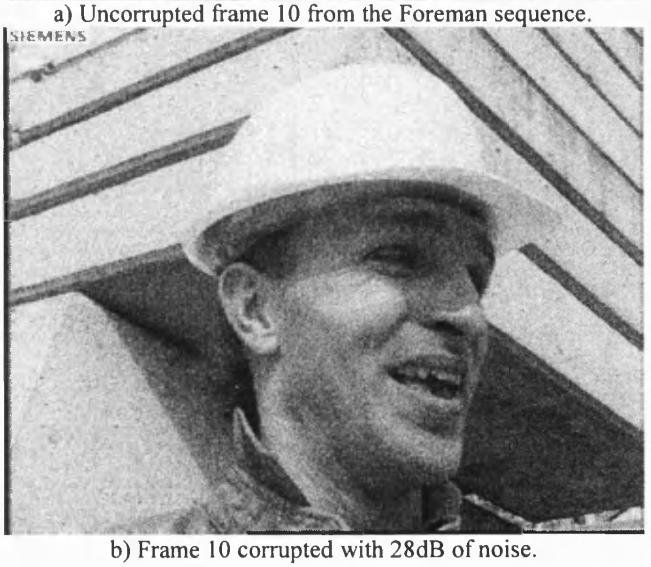


b) Filter output for the Foreman sequence.

Figure 9.6: K-NN filter performance.



a) Uncorrupted frame 10 from the Foreman sequence.

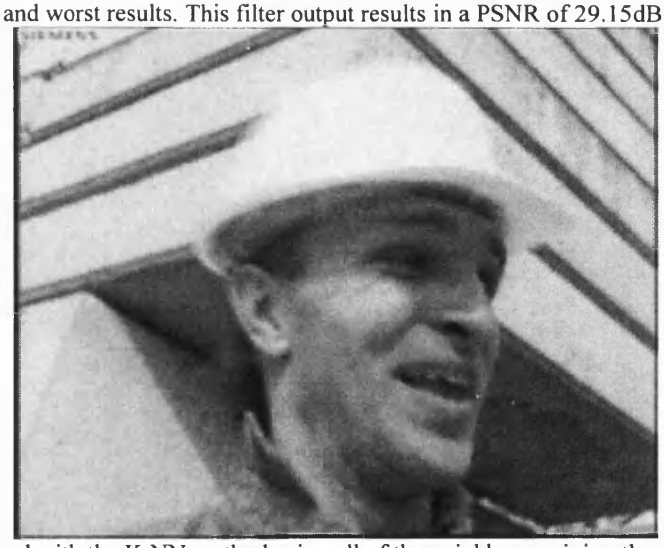


b) Frame 10 corrupted with 28dB of noise.

Figure 9.7: Original and corrupted of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.

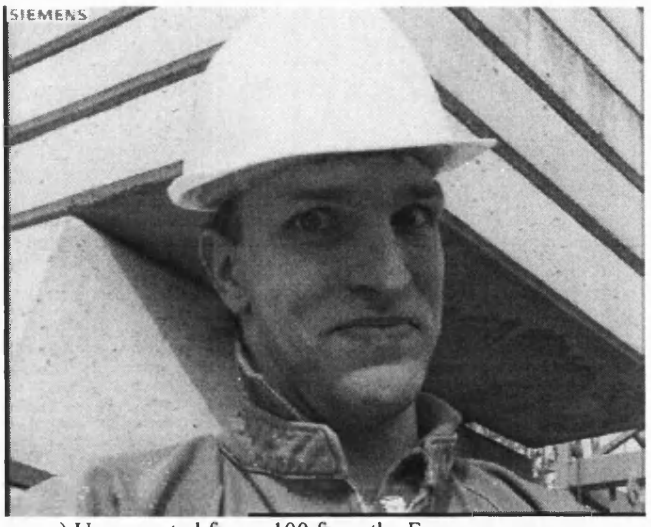
a) Frame 10 filtered with the K-NN method using 19 of the neighbours giving the best filter output resulting in a PSNR of 31.44dB.

b) Frame 10 filtered with the K-NN method using 5 of the neighbours giving a result between the best and worst results. This filter output results in a PSNR of 29.15dB.

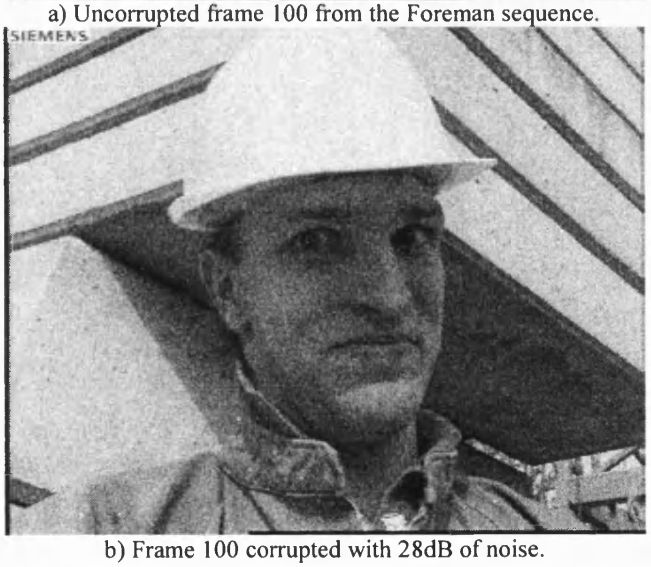


c) Frame 10 filtered with the K-NN method using all of the neighbours giving the worst filter output resulting in a PSNR of 26.66dB.

Figure 9.8: K-NN filtering of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.



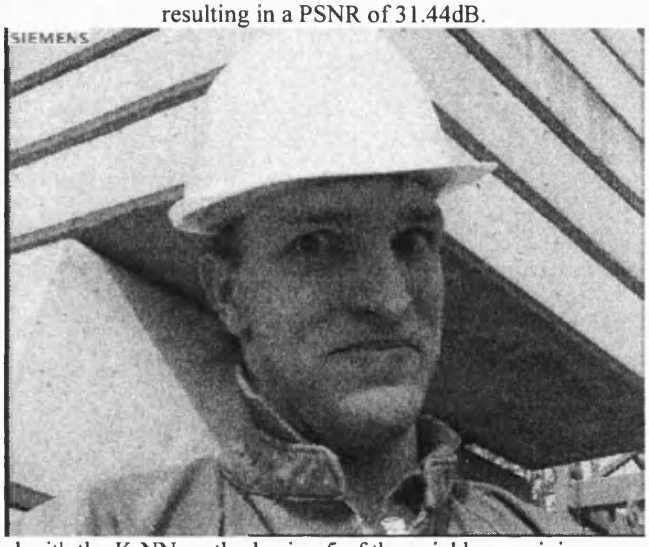
a) Uncorrupted frame 100 from the Foreman sequence.



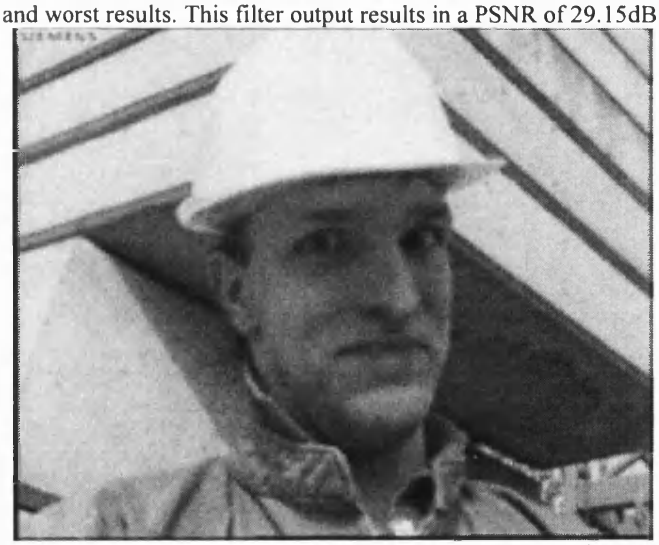
b) Frame 100 corrupted with 28dB of noise.

Figure 9.9: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.

a) Frame 100 filtered with the K-NN method using a 19 of the neighbours giving the best filter output resulting in a PSNR of 31.44dB.



b) Frame 100 filtered with the K-NN method using 5 of the neighbours giving a result between the best and worst results. This filter output results in a PSNR of 29.15dB.



c) Frame 100 filtered with the K-NN method using all of the neighbours giving the worst filter output resulting in a PSNR of 26.66dB.

Figure 9.10: K-NNfiltering of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.

9.1.3 Morphological Filtering

Before proceeding with the development of a 3D morphological system, the 2D morphological filters, namely the 2D attribute filters are used with the area and power attributes to process each frame of a sequence individually using the lossless values determined in section 7.2. The power attribute is used, as it is the best attribute at removing noise in 2D compared to area, contrast and volume. A comparison is made to the area attribute, which then gives a starting point from which to improve upon. For the 2D, the optimum attribute is found by increasing the attribute, starting from 1, and measuring the PSNR between the filtered noisy image and the original. Table 9.2 gives the optimum attribute values found (i.e. the attribute values that give the highest PSNR value). As can be seen, the 4nn connectivity provides better results than the 8nn, which was also apparent in the 2D noise removal results (see section 7.1). The power attribute shows the AF filter structure producing slightly better results than the ASF (see section 4.6.2) for both sequences. However, the area attribute only shows this for the Foreman sequence, whereas the Kiel sequence shows both the AF and ASF filter structures producing the same result.

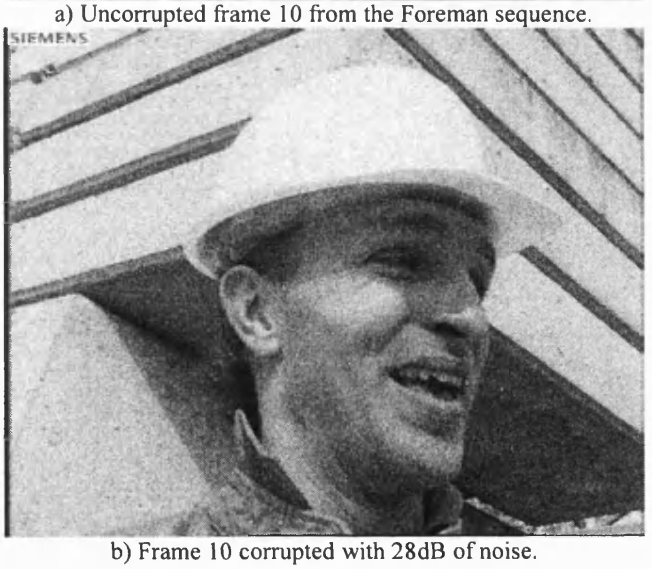
Sequence	Attribute	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Foreman 28.14dB	Area	10	32.35	10	32.32	12	31.13	12	31.13
	Power	1585	**33.06**	1585	32.96	2498	31.71	2498	31.71
Kiel 28.12dB	Area	2	29.27	2	29.27	3	28.96	3	28.96
	Power	740	**30.26**	740	30.24	841	29.89	841	29.89

Table 9.2: Optimum attribute values for spatial processing of video. Shaded cells show the best results, outlined cells show the worst results.

The input frame 10 and resultant filtered images are shown in Figures 9.11 and 9.12 and for frame 100 in Figures 9.13 and 9.14 respectively. The input images shown in Frames 9.11 and 9.13 show both uncorrupted and corrupted with 28dB of AWGN. Figures 9.12 and 9.14 show the filtered frames with the first image (Figures 9.12a and 9.14a) showing the best filter output, which is produced by using 4nn connectivity, the AF filter structure (see section 4.6.2) and the power attribute with a value of 1585. This gives a PSNR of 33.06dB when compared to the uncorrupted original sequence. This is better than the best result obtained using either the Gaussian method (see section 9.1.1), which gave a PSNR for the best combination of 32.11dB, or the K-NN method (see section 9.1.2), which gave a PSNR of 31.44dB. The second images, Figures 9.12b and 9.14b, show a result using 4nn connectivity, the ASF filter structure (see section 4.6.2) and the area attribute with a value of 10. This result is in-between, the best and worst results and gives a PSNR of 32.32dB, which is still better than that achieved by the Gaussian filtering. The last images, Figures 9.12c and 9.14c, shows the worst filter combination, which uses 8nn connectivity, the AF filter structure and the area attribute with a value of 12. The ASF filter structure also produced identical results. This combination resulted in a PSNR of 31.13dB, which is only 0.31dB worse

than the best K-NN filter combination. Visually, it is much clearer to see that noise has been removed using the morphological filters.

a) Uncorrupted frame 10 from the Foreman sequence.

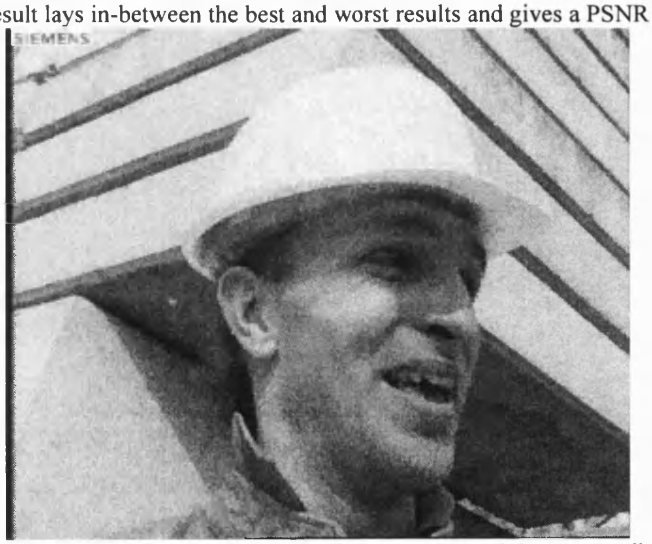


b) Frame 10 corrupted with 28dB of noise.

Figure 9.11: Original and corrupted frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.

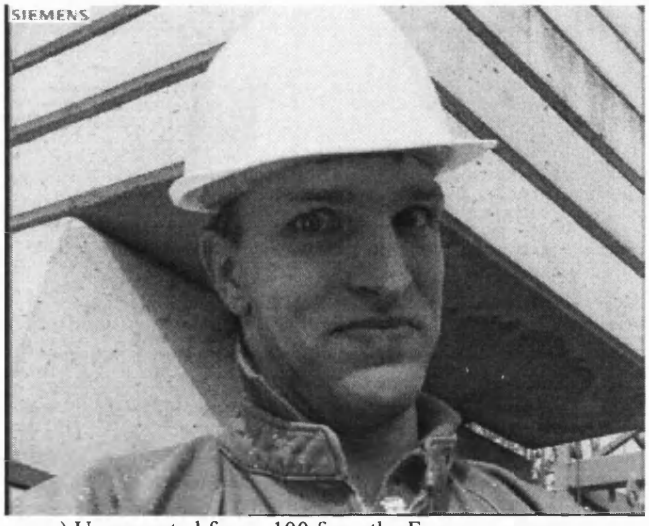
a) Frame 10 filtered using 4nn connectivity, the AF filter structure and the power attribute with a value of 1585. This is the best filter output resulting in a PSNR of 33.06dB.

b) Frame 10 filtered using 4nn connectivity, the ASF filter structure and the area attribute with a value of 10. This result lays in-between the best and worst results and gives a PSNR of 32.32dB.

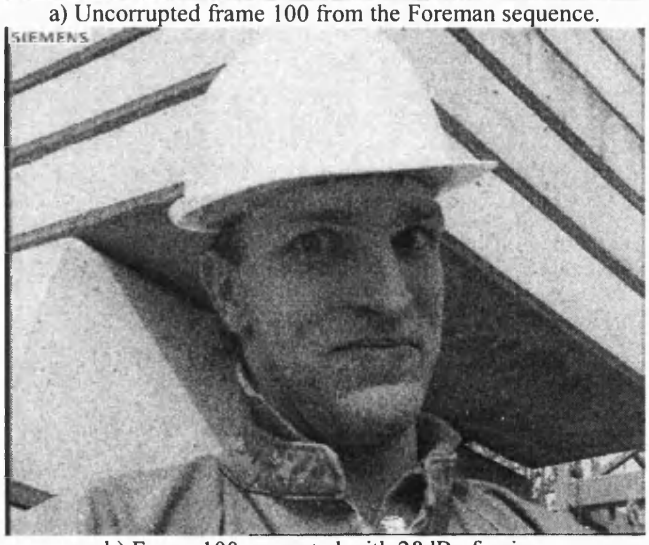


c) Frame 10 filtered using 8nn connectivity, the AF filter structure and the area attribute with a value of 12, which gives the worst result. This gives a PSNR of 31.13dB.

Figure 9.12: Morphological filtering of frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.



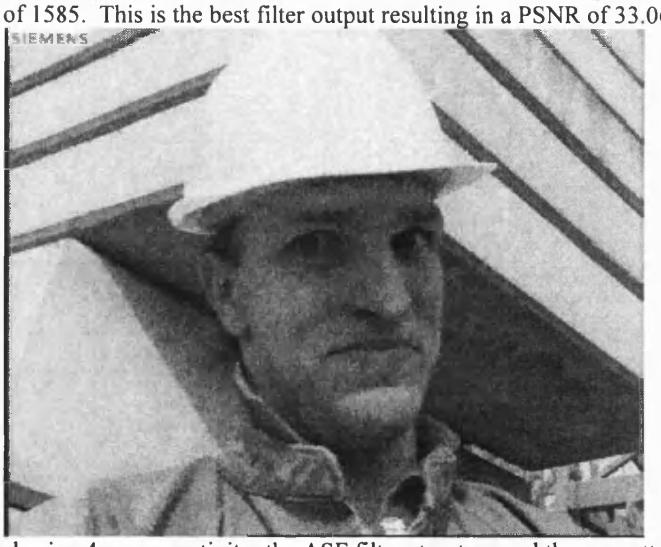
a) Uncorrupted frame 100 from the Foreman sequence.



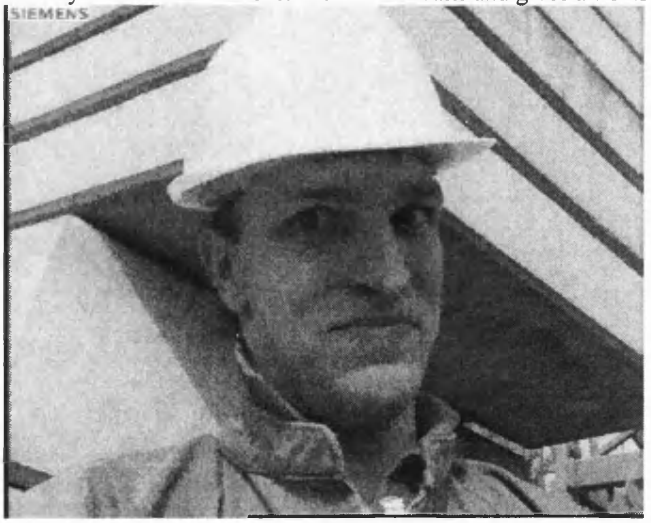
b) Frame 100 corrupted with 28dB of noise.

Figure 9.13: Original and corrupted of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.

a) Frame 100 filtered using 4nn connectivity, the AF filter structure and the power attribute with a value of 1585. This is the best filter output resulting in a PSNR of 33.06dB.



b) Frame 100 filtered using 4nn connectivity, the ASF filter structure and the area attribute with a value of 10. This result lays in-between the best and worst results and gives a PSNR of 32.32dB.



c) Frame 100 filtered using 8nn connectivity, the AF filter structure and the area attribute with a value of 12, which gives the worst result. This gives a PSNR of 31.13dB.

Figure 9.14: Morphological filtering of frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.

9.1.4 Compression Results

Although noise has been reduced using the Gaussian, K-NN and morphological spatial filters, there is no guarantee that the filtered sequences will result in better compression. Thus both sequences are preprocessed using the optimum values found for the Gaussian, K-NN and morphological filters and then compressed using the four CODEC's to a ratio of 20:1 (0.4bpp or 990Kbps). The sequences are both 352 x 288 pixels and only the y channel is used making them 8bit greyscale. In addition the sequences are played at a frame rate of 25fps so the uncompressed bandwidth is 19.34Mbps and the compressed data has a bandwidth of 990Kbps. To perform the compression tests, sequences are first filtered if required, then compressed and decompressed using a CODEC and the resultant output compared to the original uncorrupted input sequence.

Tables 9.3 and 9.4 show the output of the CODEC using the best sequences filtered using the Gaussian (see section 5.3.1.2), K-NN (see section 5.3.3) and spatial morphological (see chapter 7) filtering methods. The results show that the CODEC itself will perform some filtering. This is shown by the output of the un-processed sequences. For example, the Foreman sequence has 28.14dB of noise, but after it has been compressed and decompressed with the MJPEG CODEC, without any preprocessing, it has a PSNR of 31.32dB. The first set of results, see Table 9.3, shows that the Gaussian outperforms the K-NN filter as it did with simple noise reduction. The difference between the Gaussian filtered sequences and the K-NN filtered sequences is shown further by the Kiel sequence. For example using the Kiel sequence and the H263 CODEC, the Gaussian filtered sequence produces a PSNR of 32.66dB once compressed whereas the K-NN filter produces a PSNR of 30.68dB. All of the filter outputs do however always produce a better output than had no filtering been applied to the sequence.

Sequence (Noise level)	CODEC	PSNR (dB)		
		Un-processed	Filtered using Gaussian	Filtered using K-NN
Foreman (28.14dB)	MJPEG	31.32	**34.20**	33.77
	MPEG-II	31.24	**35.24**	35.04
	BWV	30.52	**34.77**	34.27
	H263	32.16	**36.76**	36.42
Kiel (28.12dB)	MJPEG	27.21	**29.53**	27.91
	MPEG-II	28.10	**30.71**	28.89
	BWV	27.39	**29.23**	27.87
	H263	30.25	**32.66**	30.68

Table 9.3: CODEC PSNR output at a compression ratio of 20:1 (0.4bpp or 990Kbps) using Gaussian and K-NN preprocessed sequences. Shaded cells show the best result for that particular configuration.

The second set of results, see Table 9.4, show the results for using morphological preprocessing. The filter combination that produces the best compressed sequence for the area attribute is the same as the optimum filter for noise reduction. For example, using the Foreman sequence, the optimum filter combination used 4nn connectivity and the AF filter structure for noise reduction. This combination also produces the best compressed output for all except the BWV CODEC. The Kiel sequence shows

the same for the area attribute, that is that the optimum noise reduction filter also produces the best CODEC output. However, the power attribute shows that the best CODEC output is produced mostly from the same combination that produced the best noise reduced output. The two exceptions being when the Foreman sequence is used with the MJPEG CODEC and the Kiel sequence is used with the BWV CODEC. The best result for both the Foreman and Kiel sequences are obtained with the AF filter structure, 4nn connectivity and the H263 CODEC. Thus showing that the performance of the preprocessing also depends on the CODEC used. Finally, the power attribute also produces the highest PSNR output for all filter configurations. The morphological filtering is shown to be better than both the Gaussian and K-NN filtering (see section 9.1.3). However, these results show that the Gaussian filtered sequences produce more compressible sequences, as it is the Gaussian results that have a higher PSNR, indicating that the CODEC's are storing more information in the same amount of data. This has proven that reducing noise increases the compressibility of a video sequence. Like the spatial noise reduction, the power attribute produces the best results using the morphological filtering. The majority of the results show that the 4nn connectivity results in a better output than the 8nn and that the AF is generally better than the ASF. Both sets of results show clearly that the H263 CODEC performs better than MJPEG, MPEG-II and BWV. In addition, the output of all un-processed sequences show an improvement in the PSNR over the noisy sequences. Similarly, all filtering combinations result in a further improvement on the PSNR, showing that the preprocessing improves the compressibility of an image sequence, regardless of the CODEC used.

Sequence	CODEC	Attribute	PSNR (dB)				
			Un-processed	4nn		8nn	
				AF	ASF	AF	ASF
Foreman 28.14dB	MJPEG	Area	31.32	32.57	32.56	32.44	32.44
		Power		**32.77**	32.69	32.75	32.75
	MPEG-II	Area	31.24	33.34	33.31	32.82	32.82
		Power		**33.62**	33.58	33.17	33.17
	BWV	Area	30.52	32.87	32.88	32.61	32.61
		Power		**33.03**	33.01	32.87	32.87
	H263	Area	32.16	34.27	34.24	33.77	33.77
		Power		**34.71**	34.67	34.13	34.13
Kiel 28.12dB	MJPEG	Area	27.21	27.33	27.33	27.31	27.31
		Power		**27.67**	27.54	27.55	27.55
	MPEG-II	Area	28.10	28.56	28.56	28.44	28.44
		Power		**28.99**	28.98	28.82	28.82
	BWV	Area	27.39	27.59	27.59	27.59	27.59
		Power		27.71	27.70	**27.72**	**27.72**
	H263	Area	30.25	30.53	30.53	30.31	30.31
		Power		**30.84**	30.82	30.79	30.80

Table 9.4: CODEC PSNR output at a compression ratio of 20:1 (0.4bpp or 990Kbps) using morphologically preprocessed sequences. Shaded cells show the best result for that particular configuration and the outlined cells show the worst output.

9.2 Spatio-temporal morphological filtering

Although the spatial morphological filtering methods have shown to increase the performance of a video CODEC, processing spatial introduces flicker between frames. This can be seen as either noise, which is removed in one frame and not the next, or a change in the brightness of the frame. Either way it can be annoying for an observer. Hence a spatio-temporal filtering system is required that will not only filter a frame, but will also filter adjacent frames. This will help to reduce flicker and remove more noise. Since noise is uncorrelated both spatially and temporally, it is thought that this method will allow the noise to be better isolated and hence remove the noise whilst keeping more of the frames visually intact. Extension to 3D also changes the attribute types. The are attribute actually becomes volume for example as it is spread into another dimension. However, it is still measured by counting the number of pixels in the region, so is not the same measure as the volume that was defined for spatial use (see section 6.2.2.3). Hence to keep things simple and easy to compare to, the 3D version of area is simply referred to as '**Area x time**'. Similarly the power is simply measured as '**Power x time**'.

9.2.1 Non-Overlapping Window Processing

The most obvious starting point to developing a 3D preprocessing system is to simply take the 2D method, and extend it to 3D (see chapter 4). This is a relatively simple method. Blocks of frames, much like the GOP (see section 2.3.6), are extracted and processed. The block is then moved to the next block as shown in Figure 9.15, processing the frames in a non-overlapping manner. The process is repeated until all frames have been processed. The reconstruction process and extrema region growing are modified to encompass the previous and next frames using the principles described in chapter 4. However, the connectivity has to be redefined as shown in Figure 9.16. Values from other blocks are not taken into account when processing the current block. For example, when processing block 2, even when filtering the first or last frame in the block, frames from block 1 and 3 are not used.

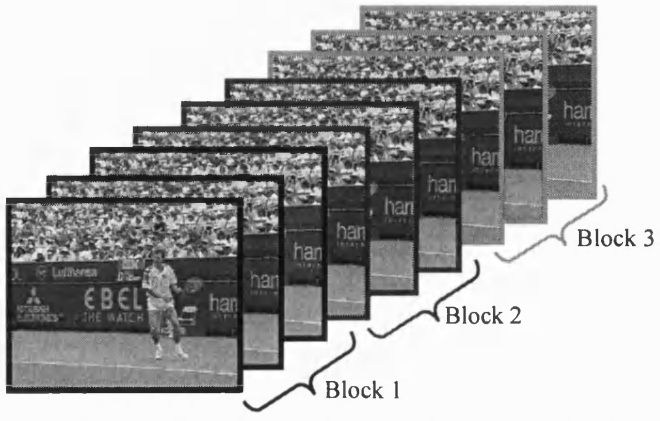


Figure 9.15: Non-overlapping spatio-temporal filtering.

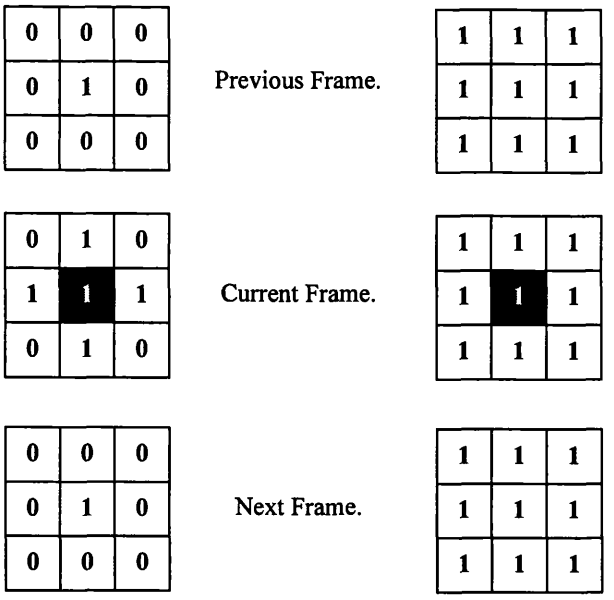


Figure 9.16: Extension of the 4nn (left) and 8nn (right) to 3D giving the 6nn (or 4_{nn}^{3D}) and 26nn (or 8_{nn}^{3D}) respectively.

The optimum results for this method are shown in Table 9.5. This shows that for the Foreman sequence, using the power attribute, 4_{nn}^{3D} connectivity, the AF filter structure and a block size of 3 produces the best performance with a PSNR increase of 4.34dB compared with an increase of 3.96dB using the area attribute with the same connectivity and filter structure. The best performing combination for the Kiel sequence using the power attribute is produced using either the AF or ASF filter structure using 4_{nn}^{3D} connectivity and a block size of 3, which gives an increase in performance of 1.98dB. However, using the area attribute, the best filter is produced using the same connectivity and both the AF and ASF filter structures with an increase of 1.14dB being achieved. Regardless of the filter combination used, an improvement on the PSNR is always achieved. Hence if real time operation is required then the fastest filter could be used knowing that it will remove some of the noise, although it may not be the best solution for noise reduction. For the 4_{nn}^{3D} connectivity, the AF filter structure results are the same as or better than those produced by the ASF filter structure. The 4_{nn}^{3D} connectivity also produces better results than the 8_{nn}^{3D} connectivity. The AF and ASF filter structures produce identical PSNR values when used with 8_{nn}^{3D} connectivity, which coincidently always produces the worst output. In addition it can be seen that the optimum results for each combination of attribute, connectivity and filter structure are obtained using a block size of 3 and the worst output is produce using a block size of 7. As the block size increases, the performance of the preprocessor drops and the attribute value rises. Thus the complexity of the preprocessor also rises with no extra gain.

Comparing these results to those using the spatial only morphological processing (see section9.1.3), it is clear that the results obtained using this method is not as good as the spatial only. For example using the spatial only method, the power attribute, 4nn connectivity, the AF filter structure and the Foreman sequence, the filtered output has a PSNR of 33.06dB. Using this non-overlapping method with the same filter combinations and a block size of 3, the filtered output has a PSNR of 32.48dB. However, when compared with the Gaussian (see section 9.1.1) and K-NN (see section 9.1.2) filtering methods, a slight gain in performance is achieved over these methods. The Gaussian method achieves a PSNR of 32.11 dB and 29.74dB (see Table 9.1) for the Foreman and Kiel sequences respectively. In addition this method has the advantage that it can reduce flicker observed between frames although some flickering is still visible between adjacent blocks.

Sequence	Attribute	Block Size	4_{nn}^{3D}				8_{nn}^{3D}			
			AF		ASF		AF		ASF	
			Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Foreman 28.142dB	Area x time	3	24	32.10	24	32.08	33	29.59	33	29.59
		5	30	31.97	30	31.96	49	29.37	49	29.37
		7	34	31.94	34	31.93	48	29.29	48	29.29
	Power x time	3	4134	**32.48**	3844	32.46	10401	29.80	10401	29.80
		5	6000	32.24	5000	32.23	15294	29.51	15803	29.51
		7	6081	32.15	6081	32.14	21117	29.41	21117	29.41
Kiel 28.12dB	Area x time	3	3	29.26	3	29.26	6	28.76	6	28.76
		5	3	29.24	3	29.24	8	28.72	8	28.72
		7	3	29.23	3	29.23	8	28.70	8	28.70
	Power x time	3	898	**30.10**	898	**30.10**	1294	29.12	1294	29.12
		5	898	30.03	898	30.03	1402	28.97	1439	28.97
		7	898	30.01	898	30.00	1295	28.94	1295	28.94

Table 9.5: Optimum attribute values and resultant PSNR for non-overlapping method. Shaded cells show the best result for that particular configuration and the outlined cells show the worst output.

9.2.1.1 Compression Results

Although this method is 3D, it will not process as much of the data as the 2D method. This is due to the relation hip between 2D and 3D extrema:

$$3D_{extrema} \subset 2D_{extrema} \tag{9.1}$$

As a result, some of the 2D extrema will not be 3D extrema. Since this method only processes 3D extrema, there will be a proportion of the 2D extrema that are not grown. However, although not as much noise is removed as is by the 2D method, it may compress better due to the temporal noise being reduced resulting in a higher correlation between adjacent frames. Thus the optimal filter values are used to preprocess the sequences, which are then compressed to a ratio of 20:1. The results are given in Table 9.6. The results all show an increase in the PSNR compared to the noisy

unprocessed sequences. With one exception, when using the Foreman sequence the AF filter structure and 4_{nn}^{3D} connectivity produce better results than the ASF filter structure. However, when 8_{nn}^{3D} connectivity is used, the ASF filter structure produces results that are better or equal to the AF filter structure with two exceptions. The Kiel sequence is less correlated showing an equal split between the filter structures that produce the best results using 4_{nn}^{3D} connectivity. When 8_{nn}^{3D} connectivity is used, the AF and ASF filter structures produce identical results with three exceptions. The 4_{nn}^{3D}, which is the 3D equivalent to the 4nn is shown to always be better than the 8_{nn}^{3D} for both sequences. This was shown to be true for both the image noise reduction (see section 7.1) and for the spatial only processing of image sequences (see section 9.1). Some of the results do show an improvement over the compression results obtained through spatial only filtering. Thus although the noise reduction was not as good as the spatial method, many of the compressed results are better. This is due to the fact that this method smoothes data temporally, allowing adjacent frames to be correlated more than is possible with spatial filtering, which allows the motion compensation part of the CODEC to find a better match than is possible with the spatial filtering. When these results are compared compression results using the Gaussian and K-NN filtering (see section 9.1.4), it can be seen that both of these systems produce more compressible sequences. The Gaussian can produce flickering results as adjacent frames are not used in calculating the output. It is debatable though as to whether they are visually better, which is looked at in section 9.4.

Sequence	CODEC	Attribute	Block Size	PSNR (dB)				
				Un-processed	4^{3D}_{nn}		8^{3D}_{nn}	
					AF	ASF	AF	ASF
Foreman 28.14dB	MJPEG	Area x time	3	31.32	32.79	32.78	31.64	31.64
			5		32.78	32.78	31.42	31.42
			7		32.80	32.80	31.38	31.38
		Power x time	3		**32.94**	**32.94**	32.07	32.07
			5		**32.94**	32.93	31.85	31.85
			7		**32.94**	32.93	31.72	31.72
	MPEG-II	Area x time	3	31.24	33.37	33.36	31.57	31.57
			5		33.37	33.36	31.32	31.32
			7		33.38	33.37	31.32	31.32
		Power x time	3		**33.62**	33.60	32.02	32.02
			5		33.53	33.53	31.80	31.80
			7		33.50	33.49	31.67	31.67
	BWV	Area x time	3	30.52	33.00	33.00	31.50	31.50
			5		33.01	33.01	31.18	31.18
			7		33.03	33.03	31.16	31.16
		Power x time	3		**33.17**	**33.17**	31.94	31.94
			5		33.12	32.12	31.64	31.64
			7		33.11	33.11	31.49	31.49
	H263	Area x time	3	32.16	34.29	34.28	32.32	32.32
			5		34.29	34.28	32.04	32.04
			7		34.30	34.30	32.07	32.08
		Power x time	3		**34.60**	34.59	32.86	32.86
			5		34.47	34.47	32.58	32.58
			7		34.45	34.44	32.43	32.43
Kiel 28.12dB	MJPEG	Area x time	3	27.21	27.46	27.46	27.37	27.37
			5		27.47	27.47	27.33	27.33
			7		27.47	27.47	27.34	27.34
		Power x time	3		**27.69**	27.56	27.50	27.50
			5		27.57	27.57	27.48	27.48
			7		27.57	27.57	27.48	27.36
	MPEG-II	Area x time	3	28.10	28.59	28.59	28.19	28.19
			5		28.59	28.59	28.11	28.11
			7		28.57	28.57	28.12	28.12
		Power x time	3		**28.93**	**28.93**	28.42	28.42
			5		28.91	28.91	28.34	28.34
			7		28.89	28.89	28.34	28.34
	BWV	Area x time	3	27.39	27.63	27.62	27.44	27.44
			5		27.64	27.64	27.38	27.38
			7		27.64	27.64	27.39	27.39
		Power x time	3		**27.72**	27.71	27.61	27.61
			5		27.71	27.70	27.57	27.56
			7		27.71	27.70	27.57	27.57
	H263	Area x time	3	30.25	30.50	30.50	29.98	29.98
			5		30.51	30.52	29.84	29.84
			7		30.49	30.50	29.86	29.86
		Power x time	3		30.87	30.87	30.34	30.34
			5		**30.88**	**30.88**	30.23	30.23
			7		30.87	30.87	30.23	30.23

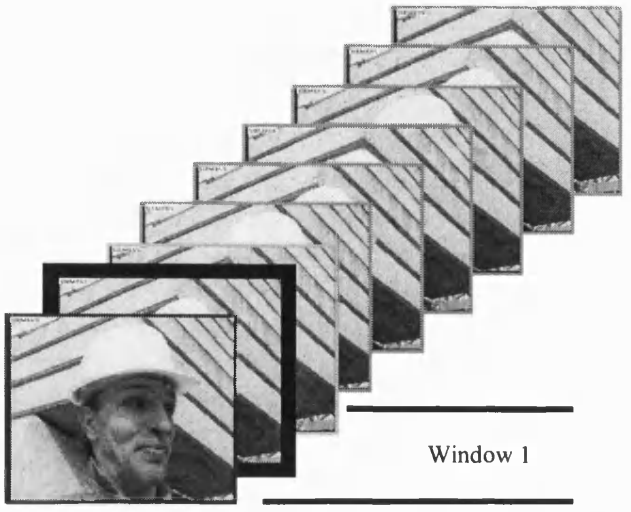
Table 9.6: CODEC PSNR output at 20:1 (0.4bpp) using non-overlapping preprocessing. Shaded cells show the best outputs and the outlined cells show the worst outputs.

9.2.2 Overlapping Window Processing

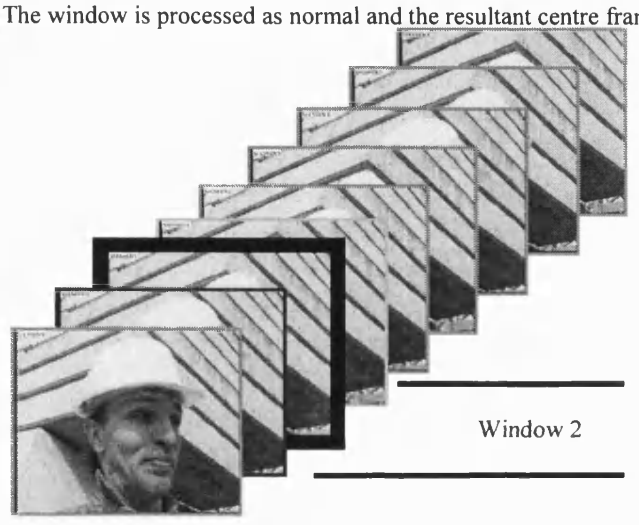
The 2D method showed that the 4nn was better than the 8nn as does the non-overlapping window method. Thus from this point forward, only the 4nn connectivity is looked at. The basic method described above is now extended to create a sliding window as shown in Figure 9.17. Each window is processed as before, but only the centre frame is saved. The window is then moved along one frame and the process is repeated until the entire sequence has been processed. There are two additional options available with this method. When the window is moved along, the previously filtered data can either be discarded or retained as the input to the next window.

The first option, throwing the previously filtered data away results in a **'Non-recursive'** filter structure. This method requires more work as the entire window has to be processed from scratch for every window. The second method, keeping the previous result is referred to as a **'Recursive'** filter structure. Once a window has been processed, the last frame is removed and the new window loaded. The window is then processed as normal, but due to the fact that much of the information has been processed before, there is less data to filter. These methods are applied to the corrupted Foreman and Kiel sequences using both the AF and ASF filter structure (see section 4.6.2) and window sizes of 3, 5 and 7. The optimum results are shown in Table 9.7, which shows for the Foreman sequence that as the window size increases, so too does the attribute value. This is due to the fact that as the window size increases, the amount of noise seen by the growing process also increases and thus a larger attribute value is required to remove this noise. However, the Kiel sequence shows the opposite. That is as the window size increases, the attribute value decreases. In addition the Foreman sequence shows that the AF filter structure performs better than the ASF filter structure although in some cases the difference is very small. For example using the non-recursive method, a window size of 7 and the area attribute. The AF filter structure results in a PSNR of 31.78dB. However the ASF filter structure gives a result of 31.77dB. Thus there is only a 0.01dB difference, which could be argued is negligible.

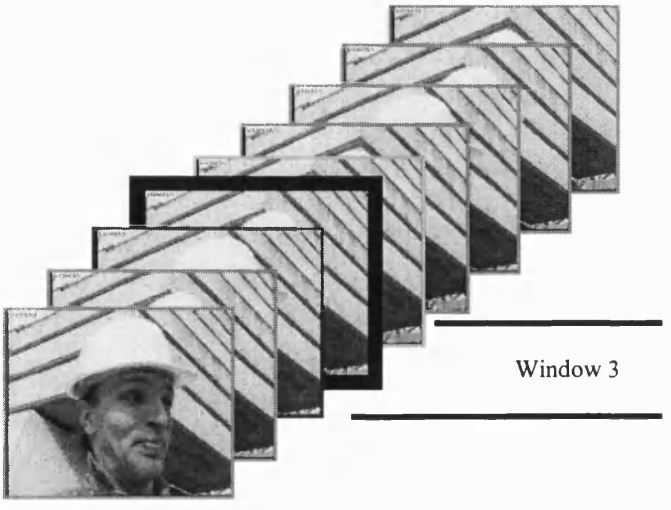
The AF filter structure gives the same or better results than the ASF filter structure for the Kiel sequence. This shows that regardless of the sequence used, the AF filter structure will always give the best results. With one exception, all of the optimum results are produced using a window size of 3, which was also the case for the non-overlapping method. In addition, the recursive method performs better than the non-recursive method for both sequences, the power and area attributes and the AF and ASF filter structures. For example, using the Foreman sequence with the power attribute, a window size of 3 and the AF filter structure, the non-recursive method results in a PSNR of 32.34dB whereas the recursive method gives a PSNR of 32.74dB. The power attribute also performs better than the area attribute for the corresponding filter combination. Comparing this method to the spatial only morphological filter (see section 9.1.3) shows that this technique provides slightly less improvement than the spatial only method, but that it performs better than both the Gaussian (see section 9.1.1) and K-NN (see section 9.1.2) filtering methods. In addition, the non-recursive method produces less of an improvement than the non-overlapping method. However, the recursive method gives an improvement upon the non-overlapping method.



a) The window is processed as normal and the resultant centre frame is saved.



b) The window is moved along by 1 frame. This is then processed again as normal and the resultant centre frame saved.



c) The window is slid continuously until all frames have been processed.

Figure 9.17: Overlapping window method.

Sequence	Attribute	Remove or keep results from previous block	Window size	4_{nn}^{3D}				8_{nn}^{3D}			
				AF		ASF		AF		ASF	
				Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Foreman 28.14dB	Area x time	Non-recursive	3	26	31.96	25	31.95	34	29.48	34	29.48
			5	36	31.81	35	31.80	50	29.25	50	29.25
			7	41	31.78	40	31.77	60	29.18	60	29.18
		Recursive	3	14	32.31	14	32.29	22	29.51	22	29.51
			5	15	32.25	15	32.24	21	29.74	21	29.74
			7	16	32.24	16	32.23	18	29.67	18	29.67
	Power x time	Non-recursive	3	4134	32.35	4090	32.32	11449	29.69	11449	29.69
			5	6188	32.09	5888	32.08	17992	29.40	17992	29.40
			7	7718	32.00	7243	31.99	25095	29.30	25095	29.30
		Recursive	3	1249	**32.74**	1296	32.73	2399	30.21	2399	30.21
			5	1152	32.58	1152	32.57	2233	29.95	2233	29.95
			7	1152	32.52	1152	32.51	2013	29.85	2013	29.85
Kiel 28.12dB	Area x time	Non-recursive	3	4	29.23	4	29.23	7	28.72	7	28.72
			5	3	29.21	3	29.21	9	28.67	9	28.67
			7	3	29.21	3	29.21	11	28.66	11	28.66
		Recursive	3	3	29.32	3	29.32	4	28.86	4	28.86
			5	3	29.33	3	29.33	4	28.84	4	28.84
			7	3	29.32	3	29.32	4	28.84	4	28.84
	Power x time	Non-recursive	3	961	30.06	1009	30.05	1505	29.06	1440	29.06
			5	961	29.97	898	29.96	1681	28.91	1681	28.91
			7	949	29.94	949	29.94	1600	28.86	1600	28.86
		Recursive	3	530	**30.15**	530	**30.15**	625	29.35	625	29.35
			5	445	30.05	448	30.04	576	29.21	576	29.21
			7	404	29.99	404	29.99	500	29.15	500	29.15

Table 9.7: Optimum attribute values and resultant PSNR for overlapping method. Shaded cells show best outputs and outlined cells show the wrost.

9.2.2.1 Compression Results

Since the spatial only (see section 9.1.3) and the spatial-temporal block (see section 9.2.1) methods showed that for most cases, 14 out of 16 (or 87.5%), the best noise reduction combination of parameters also produced the best compressed output. When this was not the case, the difference between the best CODEC output and the CODEC output using the optimum noise reduction filter parameters was marginal. In addition, all of the noise reduction results show that the 8nn (or 8_{nn}^{3D}) connectivity performs worse that the 4nn (or 4_{nn}^{3D}) connectivity. The CODEC results also show that with one exception on the spatial only method (see Table 9.4), that the 4nn connectivity performs better than the 8nn connectivity. Hence for the remainder of this chapter, only the 4nn connectivity results are compressed.

Tables 9.8 and 9.9 show the results after both the Foreman and Kiel sequences, respectively, have been compressed using 4nn (or 4_{nn}^{3D}) connectivity only. The Foreman sequence shows that the AF filter structure performs better or equal to the ASF filter structure (see section 4.6.2) for the majority of cases (85% of the results). However the Kiel sequence shows that the AF filter structure performs

better than the ASF filter structure only 69% of the time. This shows that the AF filter structure can be regarded as being better than the ASF where the compressibility of images is concerned. The Foreman sequence has an average of 0.005dB difference between the two filter structures and the Kiel sequence has only a 0.004dB difference on average. Thus although the AF filter structure generally outperforms the ASF filter structure, since the average difference between the two is so small they could be considered equal to each other. Thus the AF filter structure would be favoured due to its reduced complexity. In addition the difference in PSNR between the different window sizes is very small. For example using the Foreman sequence, the MJPEG CODEC and the power attribute with a window size of 3 and in the recursive configuration, the resultant PSNR is 33.02dB. By changing the window size to 7 the PSNR drops to 32.93dB, which is a very small decrease and probably would not be noticed visually. The compressed outputs also show the same trend as the output of the noise reduction stage. That is that the recursive method produces better results than the non-recursive, although the difference is very small. For example using the Foreman sequence, the MJPEG CODEC and the power attribute with a window size of 3 and in the recursive configuration, the resultant PSNR is 33.02dB. However using the non-recursive configuration results in a PSNR of 32.96dB, which is small and would probably not be noticeable visually. In addition the power attribute performs better than the area attribute. The noise reduction results showed that there was no improvement upon the spatial method. However the compressed output does show an improvement on the optimum filters for the majority of the recursive filter results. For example, the spatial only method, using the power attribute, 4nn connectivity, the AF filter structure and the MPEG-II CODEC, the resultant PSNR was 33.62dB. Using the same combinations and a window size of 3 with the non-recursive and recursive methods, the resultant PSNR values are 33.58dB and 33.75dB respectively. As was seen with the output of the noise reduction, as the window size increases, the improvement decreases. This is due to the fact that less of the noise is being removed, which results in the CODEC not working as efficiently as it could do.

As was seen with previous compression results (see sections 9.1.4 and 9.2.1.1), when compared to the compression results obtained using the Gaussian and K-NN filters (see section 9.1.4), this method shows less compressible sequences. For example, the Gaussian filtering using the MPEG-II CODEC produced a PSNR of 35.24dB for the Foreman sequence compared to 33.75dB, which was the best achieved by this morphological filter. However a psychovisual evaluation (see section 9.4) shows that observers tend to prefer the morphologically filtered sequences over both the Gaussian and the K-NN methods.

CODEC	Attribute	Remove or keep results from previous block	Window Size	PSNR (dB)		
				Un-processed	4_{nn}^{3D}	
					AF	ASF
MJPEG	Area x time	Non-recursive	3	31.32	32.79	32.80
			5		**32.77**	**32.77**
			7		32.79	32.79
		Recursive	3		32.84	32.84
			5		32.81	32.81
			7		32.81	32.81
	Power x time	Non-recursive	3		32.96	32.96
			5		32.88	32.88
			7		32.87	32.87
		Recursive	3		**33.02**	33.01
			5		33.00	32.99
			7		32.93	32.93
MPEG-II	Area x time	Non-recursive	3	31.24	33.32	33.32
			5		33.28	33.27
			7		33.28	33.28
		Recursive	3		33.47	33.46
			5		33.46	33.45
			7		33.46	33.45
	Power x time	Non-recursive	3		33.58	33.57
			5		33.46	33.45
			7		33.42	33.42
		Recursive	3		**33.75**	33.74
			5		33.68	33.67
			7		33.64	33.64
BWV	Area x time	Non-recursive	3	30.52	32.94	32.95
			5		32.95	32.95
			7		32.97	32.97
		Recursive	3		33.04	33.04
			5		33.05	33.05
			7		33.05	33.05
	Power x time	Non-recursive	3		33.13	33.13
			5		33.08	33.08
			7		33.07	33.07
		Recursive	3		**33.23**	33.22
			5		33.19	33.18
			7		33.16	33.16
H263	Area x time	Non-recursive	3	32.16	34.20	34.19
			5		34.19	34.19
			7		34.23	34.23
		Recursive	3		34.39	34.38
			5		34.38	34.37
			7		34.38	34.37
	Power x time	Non-recursive	3		34.51	34.50
			5		34.40	34.41
			7		34.38	34.38
		Recursive	3		**34.77**	34.76
			5		34.67	34.66
			7		34.63	34.62

Table 9.8: CODEC PSNR output at 20:1 (0.4bpp or 990Kbps) using overlapping preprocessing for the Foreman sequence corrupted with noise to 28.14dB. Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Remove or keep results from previous block	Window Size	PSNR(dB)		
				Un-processed	4^{3D}_{nn}	
					AF	ASF
MJPEG	Area x time	Non-recursive	3	27.21	27.45	27.45
			5		27.48	27.48
			7		27.48	27.48
		Recursive	3		27.44	27.44
			5		27.44	27.44
			7		27.44	27.44
	Power x time	Non-recursive	3		27.70	27.70
			5		27.58	27.58
			7		27.58	27.58
		Recursive	3		**27.68**	27.67
			5		27.67	27.67
			7		27.66	27.66
MPEG-II	Area x time	Non-recursive	3	28.10	28.57	28.58
			5		28.57	28.57
			7		28.57	28.57
		Recursive	3		28.62	28.62
			5		28.63	28.63
			7		28.63	28.63
	Power x time	Non-recursive	3		28.92	28.92
			5		28.88	28.88
			7		28.87	28.87
		Recursive	3		**28.96**	28.95
			5		28.90	28.89
			7		28.87	28.86
BWV	Area x time	Non-recursive	3	27.39	27.61	27.60
			5		27.64	27.63
			7		27.64	27.64
		Recursive	3		27.61	27.61
			5		27.62	27.62
			7		27.63	27.63
	Power x time	Non-recursive	3		**27.71**	**27.71**
			5		**27.71**	**27.71**
			7		27.70	**27.71**
		Recursive	3		27.68	27.68
			5		27.65	27.65
			7		27.65	27.65
H263	Area x time	Non-recursive	3	30.25	30.40	30.39
			5		30.51	30.52
			7		30.51	30.51
		Recursive	3		30.42	30.42
			5		30.44	30.45
			7		30.45	30.45
	Power x time	Non-recursive	3		30.86	30.84
			5		30.86	**30.89**
			7		30.87	30.87
		Recursive	3		30.83	30.82
			5		30.82	30.82
			7		30.81	30.81

Table 9.9: CODEC PSNR output at 20:1 (0.4bpp or 990Kbps) using overlapping preprocessing for the Kiel sequence corrupted with noise to 28.12dB. Shaded cells show the best results and the outlined cells show the worst.

9.3 Hybrid Spatial and Spatio-temporal Morphological Filtering

Some of the results from the two spatio-temporal methods described previously give an improvement over spatial only processing. Thus it is proposed that by combining the spatial only and the spato-temporal methods will provide the best of both worlds. There are two possible ways in which this can be done. The first is to filter using the spatial methods and then one of the spato-temporal methods in a sequential way (see section 9.3.1) and the second method processes spatial and spatio-temporally simultaneously (see section 9.3.2).

9.3.1 Combined Spatial and Spatio-Temporal Method

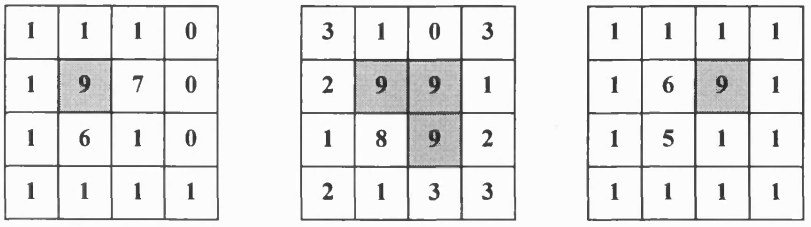
This method combines spatial and spatio-temporal processing by spatially filtering the sequence first, using a selection of filter structures and connectivity. Then the sequence is processed using one of the previously described spatio-temporal methods with the same filter structures and the corresponding 3D version of the connectivity. For the area attribute, the 3D area size must be larger than the 2D. This is because all of the extrema in the sequence have already been filtered to the 2D area size. Since all 3D extrema are also 2D extrema, the 3D extrema area size must therefore be greater than the 2D for any temporal filtering to take place, as can be seen from . If the 3D attribute had of been a value of 2 for example, the 3D growing would have done nothing. However, for the power limit, even if a flat zone has reached its limit spatially, it may not have reached the limit temporally. This is shown more clearly in Figure 9.19. Thus the temporal attribute value can be lower than the spatial limit.

The optimum attribute value for spatial processing found in section 9.1.3 is used to process the sequence spatially. Thus for the Foreman sequence using the AF filter structure and 4nn connectivity, with the non-overlapping block method of window size 3, the sequence is first processed spatially to an area of 10. The 3D optimum attribute values are then found by filtering the spatially filtered image sequence using a range of 3D attribute values. The range of values used is decreased and cantered around the current optimum value. The optimum 3D attribute values and the resultant gain in image quality are shown in Table 9.10. This shows that for the Foreman sequence, the AF filter structure outperforms the ASF filter structure for all of the filter combinations, as does the Kiel sequence when using the power attribute. However, for the area attribute and the Kiel sequence, the results show the AF and ASF filter structures producing the same results. This is due to the fact that the attribute values are small and hence the resultant filtered images will be very similar, if not identical.

Using the Foreman sequence, the spatial only filter (see section 9.1.3) gives a maximum improvement of 4.91dB (see Table 9.2), the non-overlapping spatio-temporal method gives a maximum improvement of 4.32dB (see Table 9.5). The overlapping spatio-temporal method gives a maximum increase of 4.60dB (see Table 9.7). However, this combined method has increased the maximum improvement to 5.00dB (see Table 9.10). Hence this method has shown to produce better results than all other methods, at the expense of added complexity. This is again an improvement over the

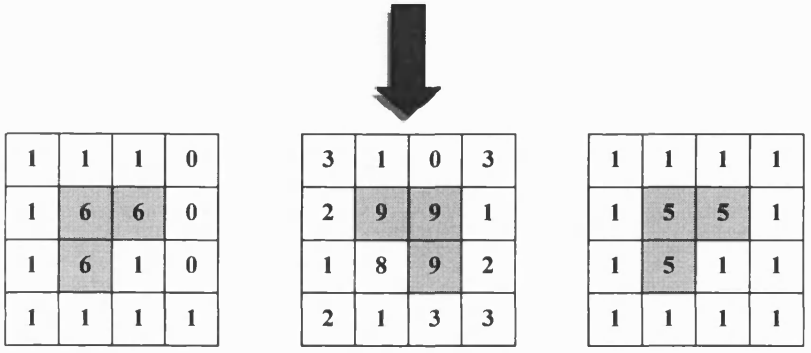
Gaussian (see section 9.1.1) and K-NN (see section 9.1.2) filtering methods, which gave an increase of 3.97dB and 1.6dB respectively for the Foreman sequence. Gains are also shown for the Kiel sequence, although they are not as good as the improvements made by the Foreman sequence.

For the area attribute, regardless of the sequence, the non-recursive is shown to produce better results than the non-overlapping method and the recursive shown to produce better results than both the non-recursive and the non-overlapping methods. However, the power attribute shows the recursive method producing better results than the non-recursive and non-overlapping methods, but that the non-overlapping method produces better results than the non-recursive method. Unlike previous results, which showed that a window size of 3 produced the optimum results, here for most cases a window size of 5 or 7 produce the optimum results. This is due to the fact that the spatial processing has removed a significant proportion of the noise prior to spatio-temporal processing. This also results in the 3D attribute values being smaller than previous methods. For example using the non-overlapping method with a window size of 3, the AF filter structure and 4_{nn}^{3D} connectivity, the optimum attribute value for area size is 20 and 242 for the power attribute. Using the non-overlapping method (see section 9.2) without spatial filtering, the area attribute value is 24 and 4134 for the power attribute. In addition, using smaller attribute values will decrease the computational time required for the 3D processing. This method has shown that by combining spatial and spatio-temporal, the performance of the preprocessing system is improved as a direct result of combining the properties of both spatial and spatio-temporal methods.

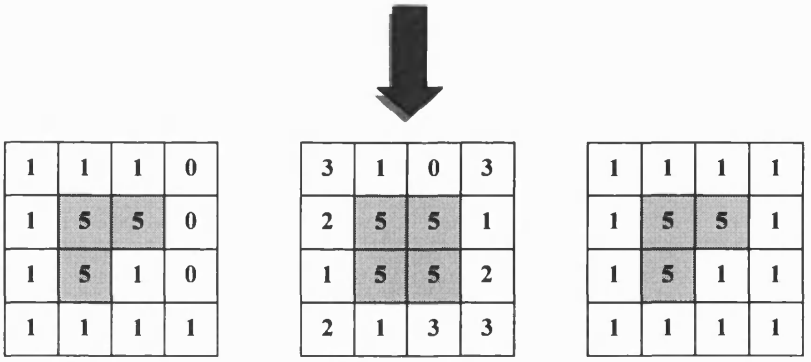


Previous Frame Current Frame Next Frame

Input sequence.

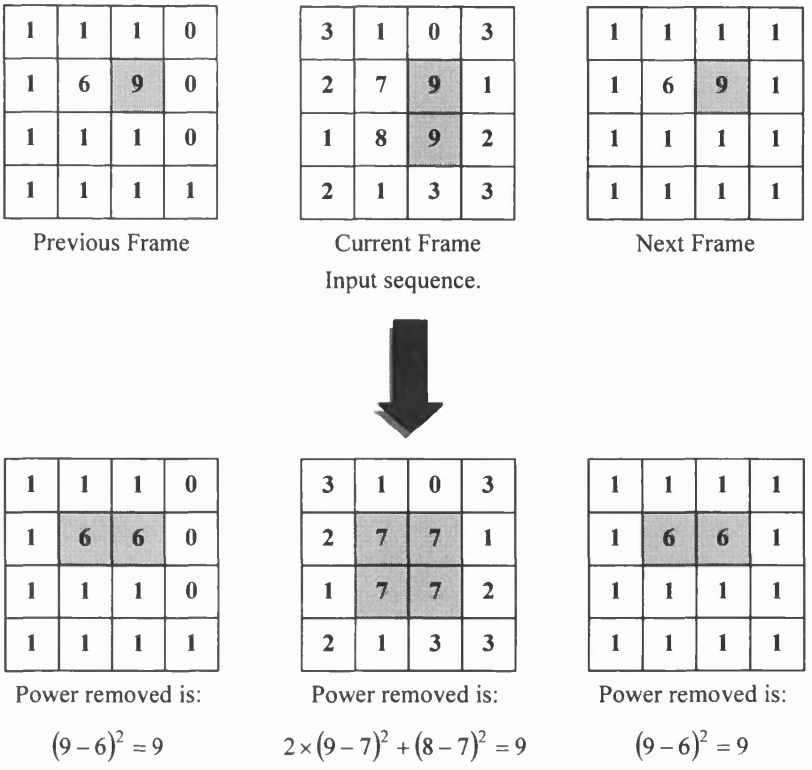


Each frame is grown spatially using the area attribute with a value of 3.



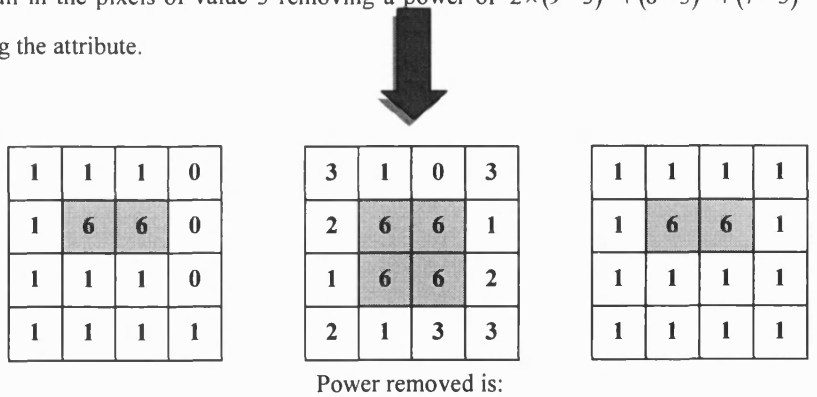
The current frame is the only one with extrema in 3D and hence is the only region to be grown in 3D. This is then grown to the 3D attribute value, in this case that is an area size of 10. Hence this pulls in the pixel on the current frame with a value of 8, and the pixels with a value of 6 and 5 on the previous and next frames respectively.

Figure 9.18: Combined spatial and spatio-temporal area attribute.



Each frame is grown spatially using the power attribute with a value of 25. The previous and next frames grow to include the pixel with the value 6. If they grew any further, they would pull in all the pixels of value 1, removing a power of $(9-1)^2+(6-1)^2 = 89$ which exceeds the attribute.

The current frame grows to encompass the pixels with value 8 and 7. If it was to grow any more, it would pull in the pixels of value 3 removing a power of $2\times(9-3)^2+(8-3)^2+(7-3)^2 = 113$ again exceeding the attribute.



Power removed is:

$$2\times(9-6)^2+(8-6)^2+(7-6)^2 = 23$$

The sequence is then processed in 3D. The current frame is the only frame with 3D extrema. This is again grown so long as the attribute is not exceeded. In this case, it grows to encompass the pixels with value 6 in the previous and next frames.

Figure 9.19: Combined spatial and spatio-temporal power limit.

Sequence	Attribute	Spatio-Temporal Method	Remove or keep results from previous block	Window size	4^{3D}_{nn} AF Attribute Value	4^{3D}_{nn} AF PSNR (dB)	4^{3D}_{nn} ASF Attribute Value	4^{3D}_{nn} ASF PSNR (dB)	8^{3D}_{nn} AF Attribute Value	8^{3D}_{nn} AF PSNR (dB)	8^{3D}_{nn} ASF Attribute Value	8^{3D}_{nn} ASF PSNR (dB)
Foreman 28.14dB	Area x time	Non-overlapping		3	20	32.43	20	32.39	24	31.17	24	31.17
				5	20	32.43	21	32.39	24	31.16	24	31.16
				7	30	32.43	30	32.39	24	31.16	24	31.16
		Overlapping	Non-recursive	3	20	32.43	20	32.39	24	31.16	24	31.16
				5	30	32.44	30	32.40	36	31.16	36	31.16
				7	30	32.44	30	32.40	36	31.16	36	31.16
			Recursive	3	12	32.47	12	32.45	15	31.13	10	31.13
				5	14	32.49	14	32.47	14	31.13	10	31.13
				7	14	32.49	14	32.46	13	31.13	10	31.13
	Power x time	Non-overlapping		3	242	33.10	294	33.05	192	31.72	192	31.72
				5	242	33.10	376	33.05	192	31.72	192	31.72
				7	242	33.10	382	33.05	191	31.72	191	31.72
		Overlapping	Non-recursive	3	242	33.10	334	33.05	238	31.72	240	31.72
				5	288	33.10	384	33.05	192	31.72	192	31.72
				7	288	33.10	426	33.05	192	31.76	192	31.76
			Recursive	3	121	**33.14**	169	33.06	225	31.76	225	31.76
				5	84	33.13	101	33.05	144	31.75	144	31.75
				7	70	33.13	82	33.05	115	31.75	115	31.75
Kiel 28.12dB	Area x time	Non-overlapping		3	3	29.34	3	29.34	4	28.98	4	28.98
				5	3	29.35	3	29.35	5	28.99	5	28.99
				7	3	29.35	3	29.35	5	28.99	5	28.99
		Overlapping	Non-recursive	3	3	29.36	3	29.36	6	29.00	6	29.00
				5	3	29.36	3	29.36	6	29.00	6	29.00
				7	3	29.36	3	29.36	6	29.00	6	29.00
			Recursive	3	3	29.34	3	29.34	4	28.97	4	28.97
				5	3	29.35	3	29.35	4	28.98	4	28.98
				7	3	29.35	3	29.35	4	28.98	4	28.98
	Power x time	Non-overlapping		3	98	30.27	123	30.26	180	29.91	180	29.91
				5	124	**30.28**	147	30.26	192	29.91	192	29.91
				7	124	**30.28**	147	30.26	180	29.91	180	29.91
		Overlapping	Non-recursive	3	147	**30.28**	178	30.27	241	29.91	241	29.91
				5	147	**30.28**	147	30.27	241	29.91	241	29.91
				7	147	**30.28**	147	30.27	241	29.91	241	29.91
			Recursive	3	36	**30.28**	35	30.26	99	29.92	99	29.92
				5	35	**30.28**	35	30.26	68	29.91	68	29.91
				7	34	**30.28**	34	30.26	71	29.91	71	29.91

Table 9.10: Optimum results for the combined spatio-temporal methods. Shaded cells show the best results and the outlined cells show the worst.

9.3.1.1 Compression Results

The results of the combined spatial and spatio-temporal method are used as the input to the four CODEC's using the Foreman (see Tables 9.11 and 9.12) and Kiel (see Tables 9.13 and 9.14) sequences. All of the results again give an increase in the PSNR output of the CODEC, compared to had there been no filtering prior to compression. The MJPEG, MPEG-II and H263 CODEC's all show that the AF filter structure (see section 4.6.2) produces a better output than the ASF filter structure for the Foreman sequence. The BWV CODEC shows that the ASF filter structure outperforms the AF filter structure most of the time. However, the average difference between the two filter structures over all of the CODEC's is only 0.026dB.

The Kiel sequence shows 57% of the results using the AF filter structure performing best and 35% showing that the AF and ASF filter structure perform equally. That is to say that the majority, 92%, of the results show that the AF filter structure is equal or better than the ASF filter structure. For the Foreman sequence, in general the recursive method performs better than the non-overlapping and the non-recursive methods. The non-recursive and non-overlapping method are not as easy to specify. For some cases the non-overlapping performs better than the non-recursive whilst in other cases this is the other way around. However, the difference between these methods is significantly smaller than with previous methods. For example, using the Foreman sequence, the MPEG-II CODEC, the AF filter structure, a window size of 3 and the power attribute, the non-overlapping method results in a PSNR of 33.832dB. The non-recursive method gives a PSNR of 33.809 and the recursive method gives 33.844dB. Using the Kiel sequence, the best performing method depends upon the CODEC and attribute used, there is no one method that is dominant. For example, using MJPEG compression and the area attribute, the non-overlapping method performs best whilst the recursive method performs worst.

Comparing this method to the spatial only morphological filtering method, the Foreman sequence shows an improvement in the performance of the preprocessing system. The Kiel sequence shows an improvement in performance over the spatial method only for the MPEG-II and BWV CODEC's. In addition the MJPEG CODEC using the area attribute shows the same relationship. The remainder of the CODEC's show a slightly lower performance than the spatial only method. The power attribute is shown to perform better than the area attribute using the same filter combinations, regardless of the CODEC and sequence used. However when compared to the compression results gained using the Gaussian and K-NN filtering methods (see section 9.1.4), it can be seen the both of these methods produce more compressible filtered sequences than this combined morphological methods despite the fact that the morphological method reduces the noise more effectively. A simple psychovisual evaluation (see section 9.4) shows that visually the morphological methods are preferred over both the Gaussian and K-NN methods.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Block Size	PSNR(dB)		
					Un-processed	4^{3D}_{nn}	
						AF	ASF
MJPEG	Area x time	Block		3	31.32	32.64	32.63
				5		32.65	32.64
				7		32.64	32.63
		Sliding	Non-recursive	3		32.59	32.57
				5		32.59	32.58
				7		32.59	32.57
			Recursive	3		32.67	32.66
				5		32.68	32.67
				7		32.69	32.67
	Power x time	Block		3		32.97	32.96
				5		32.98	32.96
				7		32.98	32.96
		Sliding	Non-recursive	3		32.91	32.89
				5		32.91	32.89
				7		32.91	32.89
			Recursive	3		**33.02**	32.98
				5		33.01	32.98
				7		**33.02**	32.93
MPEG-II	Area x time	Block		3	31.24	33.39	33.36
				5		33.39	33.36
				7		33.38	33.36
		Sliding	Non-recursive	3		33.36	33.33
				5		33.36	33.34
				7		33.36	33.33
			Recursive	3		33.41	33.39
				5		33.43	33.42
				7		33.43	33.42
	Power x time	Block		3		33.83	33.80
				5		33.83	33.79
				7		33.83	33.80
		Sliding	Non-recursive	3		33.81	33.77
				5		33.81	33.77
				7		33.81	33.77
			Recursive	3		**33.84**	33.78
				5		**33.84**	33.77
				7		**33.84**	33.78

Table 9.11: Results for the Foreman sequence using MJPEG and MPEG-II compression. Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Block Size	PSNR(dB)		
					Un-processed	4_{nn}^{3D}	
						AF	ASF
BWV	Area x time	Block		3	30.52	32.88	32.88
				5		32.88	32.89
				7		32.87	32.88
		Sliding	Non-recursive	3		32.88	32.89
				5		32.89	32.90
				7		32.88	32.89
			Recursive	3		32.91	32.91
				5		32.93	32.92
				7		32.94	32.93
	Power x time	Block		3		33.20	33.22
				5		33.20	**33.23**
				7		33.20	**33.23**
		Sliding	Non-recursive	3		33.20	33.22
				5		33.20	33.22
				7		33.20	33.22
			Recursive	3		33.22	33.19
				5		33.20	33.19
				7		33.21	33.19
H263	Area x time	Block		3	32.16	34.30	34.28
				5		34.31	34.28
				7		34.31	34.28
		Sliding	Non-recursive	3		34.29	34.26
				5		34.30	34.27
				7		34.29	34.26
			Recursive	3		34.34	34.32
				5		34.37	34.34
				7		34.36	34.35
	Power x time	Block		3		34.89	34.85
				5		34.89	34.85
				7		34.89	34.85
		Sliding	Non-recursive	3		34.86	34.83
				5		34.86	34.83
				7		34.87	34.83
			Recursive	3		**34.91**	34.83
				5		34.88	34.82
				7		34.89	34.82

Table 9.12: Results for he Foreman sequence using BWV and H263 compression. Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Window Size	PSNR(dB)		
					Un-processed	4_{nn}^{3D}	
						AF	ASF
MJPEG	Area x time	Block		3	27.21	27.42	27.42
				5		27.43	27.43
				7		27.43	27.43
		Sliding	Non-recursive	3		27.43	27.43
				5		27.43	27.43
				7		27.43	27.43
			Recursive	3		27.41	27.41
				5		27.41	27.41
				7		27.41	27.41
	Power x time	Block		3		**27.66**	27.65
				5		**27.66**	**27.66**
				7		**27.66**	27.65
		Sliding	Non-recursive	3		**27.66**	27.65
				5		**27.66**	**27.66**
				7		**27.66**	**27.66**
			Recursive	3		**27.66**	27.65
				5		**27.66**	27.65
				7		**27.66**	27.65
MPEG-II	Area x time	Block		3	28.10	28.62	28.62
				5		28.62	28.62
				7		28.63	28.63
		Sliding	Non-recursive	3		28.60	28.60
				5		28.61	28.61
				7		28.61	28.61
			Recursive	3		28.63	28.63
				5		28.63	28.63
				7		28.63	28.63
	Power x time	Block		3		29.00	28.99
				5		29.00	28.99
				7		**29.01**	29.00
		Sliding	Non-recursive	3		29.00	29.00
				5		29.00	29.00
				7		29.00	28.99
			Recursive	3		29.00	28.98
				5		29.00	28.99
				7		29.00	28.99

Table 9.13: Results for the Kiel sequence using MJPEG and MPEG-II compression. Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Window Size	PSNR(dB)		
					Un-processed	4^{3D}_{nn}	
						AF	ASF
BWV	Area x time	Block		3	27.39	27.61	27.61
				5		27.62	27.62
				7		27.61	27.61
		Sliding	Non-recursive	3		27.63	27.63
				5		27.63	27.63
				7		27.63	27.63
			Recursive	3		27.61	27.61
				5		27.62	27.62
				7		27.62	27.62
	Power x time	Block		3		27.70	27.70
				5		27.71	27.70
				7		27.71	27.70
		Sliding	Non-recursive	3		27.71	27.70
				5		27.71	27.70
				7		27.71	27.70
			Recursive	3		27.71	27.71
				5		27.72	27.71
				7		**27.72**	**27.72**
H263	Area x time	Block		3	30.25	30.38	30.38
				5		30.40	30.40
				7		30.40	30.40
		Sliding	Non-recursive	3		30.44	30.44
				5		30.44	30.44
				7		30.44	30.44
			Recursive	3		30.34	30.34
				5		30.36	30.35
				7		30.36	30.36
	Power x time	Block		3		30.81	30.79
				5		30.80	30.79
				7		30.81	30.79
		Sliding	Non-recursive	3		30.81	30.79
				5		30.81	30.80
				7		30.80	30.79
			Recursive	3		**30.82**	30.80
				5		30.80	30.79
				7		30.80	30.79

Table 9.14: Results for the Kiel sequence using BWV and H263 compression. Shaded cells show the best results and the outlined cells show the worst.

9.3.2 Simultaneous Spatial and Spatio-Temporal Method

The previous method involved first processing the sequence spatially and then spatio-temporally. Some of these extrema will have neighbours in the temporal plane that are closer to it than those on the spatial plane. A further improvement is proposed by combining the spatial and spatio-temporal methods, as in the previous method but in a way such that they work simultaneously. The basic algorithm for this method is shown in Listing 9.1 and is based on the pixel queue method (see section **6.1**). Firstly all 2D and 3D extrema are extracted from the current window and stored in four lists. On list for 2D maxima, one for 2D minima, one for 3D maxima and one for 3D minima. Secondly, any pixels belonging to both a 2D and 3D region are removed from the 2D region list. Thus the intersection between any these sets of regions should be empty. Then starting from the first attribute value, for area size this would be a value of 1, all 2D maxima regions with the same attribute value are then grown by a size of 1. This is accomplished by finding the next highest value in the neighbours and adding it to the current region if its value is the same or lower than the current regions, other wise the next region is processed. The pixel is then added to the current region and the value of the current region is changed to that of the new pixel. This is repeated until all 2D maxima of size 1 have been processed and is then repeated with the 3D maxima, the 2D minima and then the 3D minima. The minima's work in exactly the same way as the maximas with the exception that they look for the lowest value in the neighbours. Once this is done, the area size is incremented, to 2 for example, and the process is repeated. This is repeated until a target attribute size is reached.

So far all of the results shown have consistently shown that the AF filter structure performs better than or equal to the ASF filter structure. Thus this section uses only the 4nn and 4_{nn}^{3D} connectivity and the AF filter structure. Since some of the 2D extrema are now 3D extrema, the previous optimum values found for the spatial-only filtering can no longer be used. Thus the 2D and 3D attribute values are found by filtering over a range of attribute values and slowly narrowing the range down. The optimum attribute values for this method are given in Table 9.15. The results all show an improvement in noise reduction. Compared to other methods, the attribute values used are generally smaller. For example, using the Foreman sequence, the non-overlapping method and the area attribute with a window size of 3, this method obtains optimum attribute values of 5 and 23 for 2D and 3D respectively. The spatial only method (see section 9.1) using the same combinations uses a 2D attribute value of 10. The spatio-temporal non-overlapping method (see section 9.2) uses a 3D attribute value of 24.

1. Extract all 3D regional minima ($\min_{3D}$) and maxima ($\max_{3D}$).
2. Extract all 2D regional minima ($\min_{2D}$) and maxima ($\max_{2D}$).
3. Renove all 2D extrema that area also 3D extrema;
    a. $\min_{2D} \leftarrow \min_{2D} \notin \min_{3D}$
    b. $\max_{2D} \leftarrow \max_{2D} \notin \max_{3D}$
4. Let $\lambda = 1$.
5. Whle $\lambda \leq$ target attribute value do:
    a. For all $\max_{2D}$ where $attribute(\max_{2D}) = \lambda$, do the following:
        i. Find the highest greyscale valued 2D neighbour (n) of the region,
        ii. If the value of n is greater than the value of the maxima, then the region is no longer a maxima so move onto the next maxima (go to step a),
        iii. Add pixel n to the current region,
        iv. Set the value of all pixels belonging to region to be the value of n,
    b. For all $\max_{3D}$ where $attribute(\max_{3D}) = \lambda$, do the following:
        i. Find the highest greyscale valued 3D neighbour (n) of the region,
        ii. If the value of n is greater than the value of the maxima, then the region is no longer a maxima so move onto the next maxima (go to step b),
        iii. Add pixel n to the current region,
        iv. Set the value of all pixels belonging to region to be the value of n,
    c. For all $\min_{2D}$ where $attribute(\min_{2D}) = \lambda$, do the following:
        i. Find the lowest greyscale valued 2D neighbour (n) of the region,
        ii. If the value of n is greater than the value of the minima, then the region is no longer a minima so move onto the next minima (go to step c),
        iii. Add pixel n to the current region,
        iv. Set the value of all pixels belonging to region to be the value of n,
    d. For all $\min_{3D}$ where $attribute(\min_{3D}) = \lambda$, do the following:
        i. Find the lowest greyscale valued 3D neighbour (n) of the region,
        ii. If the value of n is greater than the value of the minima, then the region is no longer a minima so move onto the next minima (go to step d),
        iii. Add pixel n to the current region,
        iv. Set the value of all pixels belonging to region to be the value of n,
    e. $\lambda \leftarrow \lambda + 1$

Listing 9.1: Simultaneous Spatial and Spatio-temporal method.

As seen throughout this chapter, the power attribute produces better results than the area attribute, regardless of the sequence used. For the Foreman sequence, the recursive method shows to produce the best results with the non-recursive method producing the worst. However, using the Kiel sequence, the recursive method shows to be the worst, the best method being the non-recursive for the area attribute and the non-overlapping for the power attribute.

Comparing this method to the spatial only method shows that for the Foreman sequence and for both the area and power attributes, only the recursive method produces results that are better. However, the Kiel sequence shows that all methods for the area attribute produce better results than the spatial only method. The power attribute though only shows 2 results that are better than the spatial only. They both use a window size of 3 with the non-overlapping and the recursive methods. Hence this method can be said to be better than the spatial method so long as a window size of 3 and the recursive method are used.

9.3.2.1 Compression Results

The optimum filtered sequences are compressed and decompressed using the four CODEC's (MJPEG, MPEG-II, BWV and H263) to evaluate the improvement in performance gained by reducing the noise present in the sequences. Tables 9.16 and 9.17 show the CODEC outputs using the optimum filter results as inputs and the four CODEC's for the Foreman sequence. Tables 9.18 and 9.19 show the results for the Kiel Sequence. For the majority of the results, this method shows an improvement over the spatial morphological method (see section 9.1.4) when using the area attribute, but the compression results of both the Gaussian and K-NN (see section 9.1.4) show that they produce more compressible results. However most of the results for the power attribute are worse than the spatial morphological method. In addition, the power attribute results using the non-overlapping and the recursive methods do perform better than the area attribute using the same filter combinations. Hence the noise reduction showed a higher increase in quality than other methods, but that the result is not as compressible as other methods. However simple psychovisual testing (see section 9.4) has shown that this morphological filtering is preferred over the Gaussian as the morphological filter produces smoother, less blurred sequences and that there is no noticeable flickering between frames. Despite this, the results show that for the most part, filtering increases the compressibility of a sequence, although not as much as other methods do.

	Attribute	Spatio-temporal method	Moving method	Block Size	4nn / 4^{3D}_{nn} AF Attribute Value 2D	4nn / 4^{3D}_{nn} AF Attribute Value 3D	PSNR (dB)
Foreman 28.14dB	Area x time	Non-overlapping		3	5	23	32.31
				5	5	30	32.27
				7	4	34	32.26
		Overlapping	Non-recursive	3	5	24	32.21
				5	5	33	32.16
				7	5	35	32.16
			Recursive	3	5	15	32.55
				5	4	16	32.56
				7	4	16	32.56
	Power x time	Non-overlapping		3	533	3844	32.70
				5	580	5090	32.56
				7	542	5386	32.51
		Overlapping	Non-recursive	3	584	4199	32.61
				5	629	5857	32.47
				7	600	5857	32.10
			Recursive	3	243	905	**33.20**
				5	125	842	33.14
				7	82	842	33.09
Kiel 28.12dB	Area x time	Non-overlapping		3	2	3	29.36
				5	2	3	29.37
				7	2	3	29.37
		Overlapping	Non-recursive	3	2	3	29.37
				5	2	3	29.37
				7	2	3	29.37
			Recursive	3	2	3	29.34
				5	2	3	29.34
				7	2	3	29.34
	Power x time	Non-overlapping		3	345	898	30.22
				5	344	898	30.20
				7	404	898	30.19
		Overlapping	Non-recursive	3	344	981	30.22
				5	370	898	30.18
				7	370	898	30.17
			Recursive	3	145	530	**30.25**
				5	97	401	30.17
				7	65	442	30.11

Table 9.15: Optimum results for the simultaneous spatio-temporal method. Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Block Size	PSNR(dB)	
					Un-processed	4^{3D}_{nn} AF
MJPEG	Area x time	Non-overlapping		3	31.32	32.77
				5		32.80
				7		32.76
		Overlapping	Non-recursive	3		32.76
				5		32.72
				7		32.74
			Recursive	3		32.74
				5		32.78
				7		32.78
	Power x time	Non-overlapping		3		32.93
				5		32.92
				7		32.92
		Overlapping	Non-recursive	3		31.34
				5		31.33
				7		31.28
			Recursive	3		**32.98**
				5		32.86
				7		32.77
MPEG-II	Area x time	Non-overlapping		3	31.238	33.46
				5		33.48
				7		33.50
		Overlapping	Non-recursive	3		33.42
				5		33.42
				7		33.43
			Recursive	3		33.53
				5		33.55
				7		33.55
	Power x time	Non-overlapping		3		33.72
				5		33.68
				7		33.65
		Overlapping	Non-recursive	3		31.78
				5		31.75
				7		31.74
			Recursive	3		**33.83**
				5		33.63
				7		33.42

Table 9.16: Simultaneous results for the Foreman sequence using the MJPEG and MPEG-II CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Block Size	PSNR(dB)	
					Un-processed	4_{nn}^{3D} AF
BWV	Area x time	Non-overlapping		3	30.52	33.06
				5		33.09
				7		33.11
		Overlapping	Non-recursive	3		33.03
				5		33.05
				7		33.07
			Recursive	3		33.04
				5		33.07
				7		33.07
	Power x time	Non-overlapping		3		33.25
				5		33.24
				7		33.21
		Overlapping	Non-recursive	3		31.38
				5		31.37
				7		31.37
			Recursive	3		**33.16**
				5		32.97
				7		32.77
H263	Area x time	Non-overlapping		3	32.16	34.43
				5		34.43
				7		34.45
		Overlapping	Non-recursive	3		34.35
				5		34.35
				7		34.38
			Recursive	3		34.50
				5		34.53
				7		34.54
	Power x time	Non-overlapping		3		34.77
				5		34.69
				7		34.65
		Overlapping	Non-recursive	3		32.39
				5		32.34
				7		32.34
			Recursive	3		**34.89**
				5		34.63
				7		34.37

Table 9.17: Simultaneous results for the Foreman sequence using the BWV and H263 CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Window Size	PSNR(dB)	
					Un-processed	4^{3D}_{nn} AF
MJPEG	Area x time	Non-overlapping		3	27.21	27.44
				5		27.45
				7		27.45
		Overlapping	Non-recursive	3		27.46
				5		27.46
				7		27.46
			Recursive	3		27.42
				5		27.42
				7		27.42
	Power x time	Non-overlapping		3		27.69
				5		**27.70**
				7		27.56
		Overlapping	Non-recursive	3		27.30
				5		27.20
				7		27.20
			Recursive	3		27.63
				5		27.57
				7		27.50
MPEG-II	Area x time	Non-overlapping		3	28.10	28.63
				5		28.63
				7		28.63
		Overlapping	Non-recursive	3		28.63
				5		28.63
				7		28.63
			Recursive	3		28.63
				5		28.63
				7		28.63
	Power x time	Non-overlapping		3		**28.99**
				5		28.98
				7		28.96
		Overlapping	Non-recursive	3		28.40
				5		28.38
				7		28.38
			Recursive	3		28.95
				5		28.79
				7		28.68

Table 9.18: Simultaneous results for the Kiel sequence for the MJPEG and MPEG-II CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst.

CODEC	Attribute	Block or Sliding	Remove or keep results from previous block	Window Size	PSNR(dB)	
					Un-processed	4_{nn}^{3D} AF
BWV	Area x time	Non-overlapping		3	27.39	27.63
				5		27.64
				7		27.64
		Overlapping	Non-recursive	3		27.64
				5		27.64
				7		27.64
			Recursive	3		27.61
				5		27.62
				7		27.62
	Power x time	Non-overlapping		3		27.74
				5		27.72
				7		**27.73**
		Overlapping	Non-recursive	3		26.71
				5		26.70
				7		26.71
			Recursive	3		27.64
				5		27.53
				7		27.42
H263	Area x time	Non-overlapping		3	30.25	30.45
				5		30.48
				7		30.50
		Overlapping	Non-recursive	3		30.52
				5		30.52
				7		30.52
			Recursive	3		30.37
				5		30.37
				7		30.37
	Power x time	Non-overlapping		3		30.88
				5		**30.89**
				7		30.89
		Overlapping	Non-recursive	3		30.03
				5		30.04
				7		30.05
			Recursive	3		30.70
				5		30.58
				7		30.37

Table 9.19: Simultaneous results for the Kiel sequence for the BWV and H263 CODEC's at a compression ratio of 20:1 (0.4bpp or 990Kbps). Shaded cells show the best results and the outlined cells show the worst.

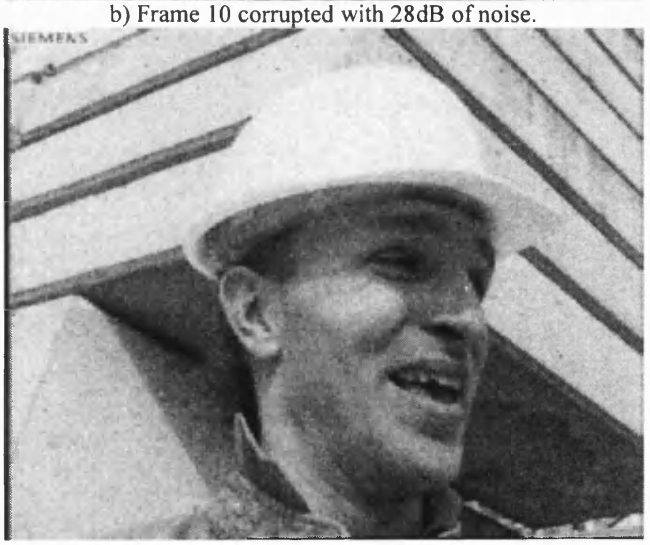
9.3.2.2 Visual Evaluation of the Simultaneous Morphological Method

The best noise reduction method is the simultaneous 2D and 3D filtering method (see section 9.3.2) To give an indication of how this affect the sequences, frames 10 and 100 have been extracted in Figures 9.20 – 9.23 to show how the image is affected by filtering to reduce the noise. The original and noise corrupted frame 10 and 100 are shown in Figures 9.20 and 9.22 respectively, where the corrupted frames have been degraded with 28.14dB of AWGN (see section 5.1.1). Although this is a low amount of noise, as seen from Figures 9.20b and 9.22b, it is still clearly visible. The goal of noise reduction is to remove as much of the noise as possible. In the ideal case this would result in the filtered image being identical to the original images (see Figures 9.20a and 9.22a).

Figures 9.21 and 9.23 show the result of filtering the noisy sequence with 3 different morphological filtering combinations, with frames 10 and 100 being shown respectively. The first images, Figures 9.21a and 9.23a, show the best filter combination, which consists of the AF filter structure (see section 4.6.2), 4nn (and 4_{mn}^{3D}) connectivity, the recursive filtering method (see section 9.2.2), a window size of 3 and the power attribute. Attribute values for 2D and 3D were found to produce the best output using values of 243 and 905 respectively. This resulted in a PSNR of 33.20dB, which is an increase of 5.06dB. Noise can be seen to be decreased by a significant amount compared to the original noisy frames (see Figures 9.20b and 9.22b). However, although flat areas can be seen on individual frames, when viewed as a sequence, these zones are hard to detect. Using a filter that is not the optimum, for example using a non-recursive method (see section 9.2.2) instead of a recursive with 2D and 3D power attribute values of 584 and 4199 respectively, still reduces noise effectively. The results of this filter are shown in Figures 9.21b and 9.23b and give a resultant PSNR of 32.61. As can be seen, although the PSNR performance is not as good as the previous method, there is little visual difference between this combination and the previous and an increase of 4.47dB is still achieved. The final images shown in Figures 9.21c and 9.23c show the worst filter combination, which again uses the non-recursive method and a window size of 7 with 2D and 3D power attribute values of 600 and 5857, still reduces the noise effectively, although visually difference can be seen between this and the best filter output. This filter combination produces a PSNR of 32.10dB, which is a 3.96dB increase. Hence even the worst filter combination still removes a significant amount of noise, which is comparable to that removed by the spatial 2D Gaussian method (see section 9.1.1) shown in Figures 9.20c, 9.22c, which produce a PSNR of 32.11dB. This has shown that the morphological noise reduction still produces visually acceptable results.

a) Uncorrupted frame 10 from the Foreman sequence.

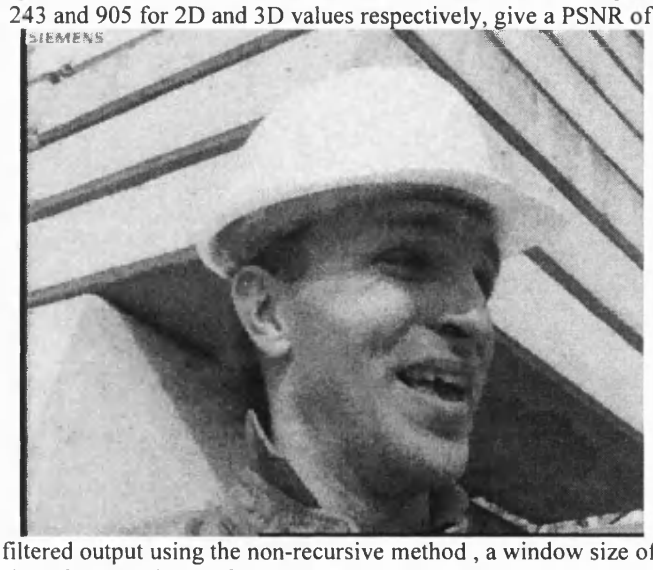
b) Frame 10 corrupted with 28dB of noise.



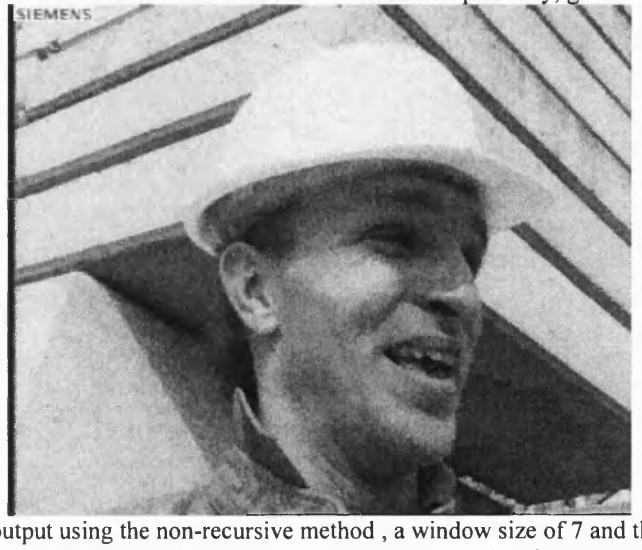
c) Frame 10 filtered with the best Gaussian method using a mask size of 3 x 3 and a σ of 0.6. the output resulting in a PSNR of 32.11dB.

Figure 9.20: The original, corrupted (uncompressed) and Gaussian frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.

a) Best filtered output using the recursive method, a window size of 3 and the power attribute with a value of 243 and 905 for 2D and 3D values respectively, give a PSNR of 33.20dB.



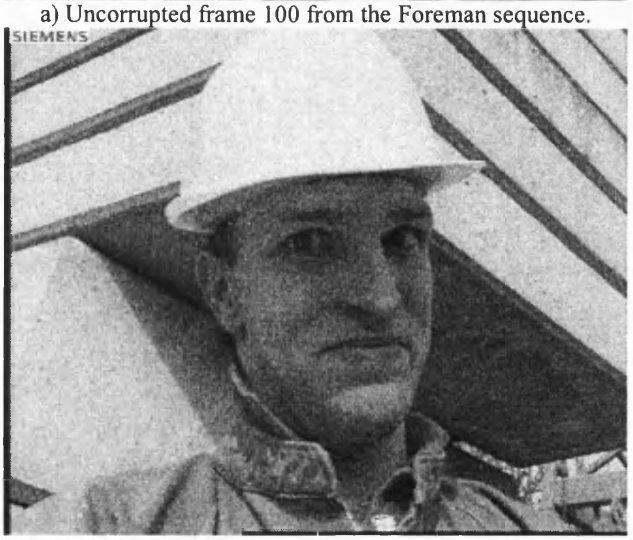
b) Intermediate filtered output using the non-recursive method , a window size of 3 and the power attribute with a value of 584 and 4199 for 2D and 3D values respectively, give a PSNR of 32.61dB.



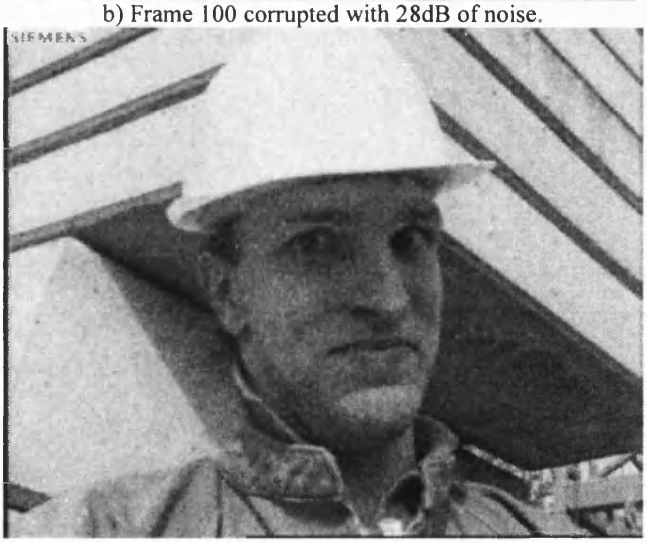
c) Worst filtered output using the non-recursive method , a window size of 7 and the power attribute with a value of 600 and 5857 for 2D and 3D values respectively, give a PSNR of 32.10dB.

Figure 9.21: Simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.

a) Uncorrupted frame 100 from the Foreman sequence.

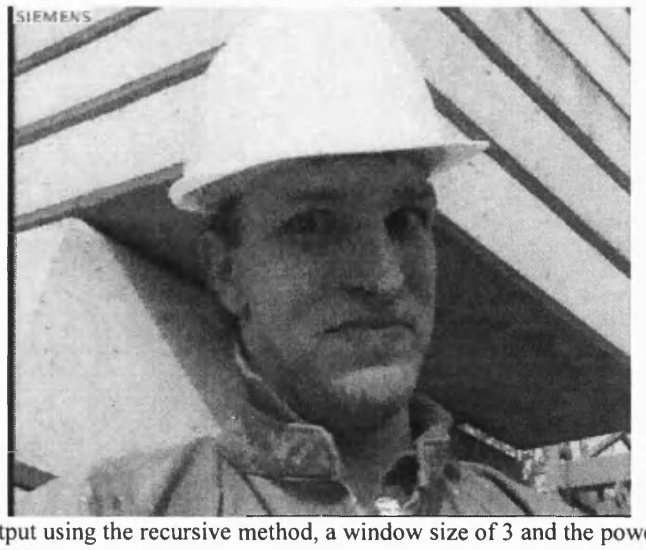


b) Frame 100 corrupted with 28dB of noise.

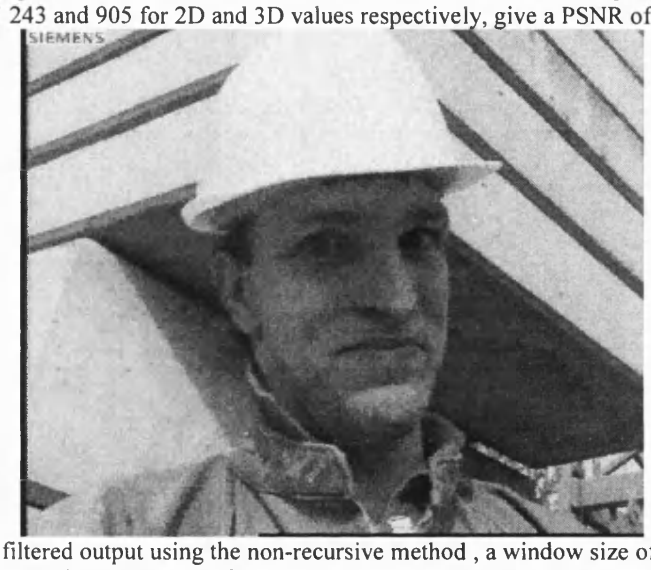


c) Frame 100 filtered with best the Gaussian method using a mask size of 3 x 3 and a σ of 0.6. The output resulting in a PSNR of 32.11dB.

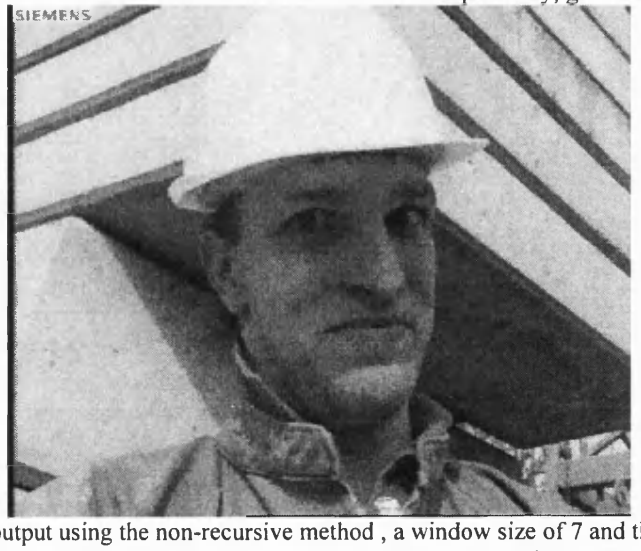
Figure 9.22: The original, corrupted (uncompressed) and Gaussian frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence.



a) Best filtered output using the recursive method, a window size of 3 and the power attribute with a value of 243 and 905 for 2D and 3D values respectively, give a PSNR of 33.20dB.



b) Intermediate filtered output using the non-recursive method , a window size of 3 and the power attribute with a value of 584 and 4199 for 2D and 3D values respectively, give a PSNR of 32.61dB.



c) Worst filtered output using the non-recursive method , a window size of 7 and the power attribute with a value of 600 and 5857 for 2D and 3D values respectively, give a PSNR of 32.10dB.

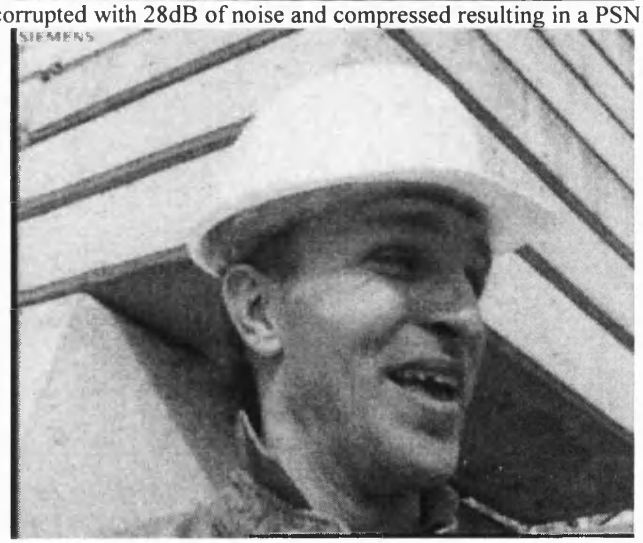
Figure 9.23: Simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence.

Resulting images from the compression of the filtered images (see Figures 9.20 – 9.23) using the H263 CODEC with a compression ratio of 20:1 (0.4bpp or 990Kbps) are shown in Figures 9.24 – 9.27. The original images for frames 10 and 100 are shown in Figures 9.24a and 9.26a, which when compared to the original uncompressed versions (see Figures 9.20a and 9.22a), show little degradation in image quality. However, the noisy images shown in Figures 9.24b and 9.26b do show coding artefacts, such as block edges. Thus this shows that noise can significantly degrade the performance of a CODEC. The uncorrupted sequence, when compressed and decompressed has a PSNR of 40.54dB when compared to the original uncorrupted input. As expected, the noisy sequence degrades this and has a PSNR of 30.25dB when compared back to the original uncorrupted sequence. This shows that noise degrades the quality of a compression system, but also since the noisy input sequence had a PSNR compared to the original uncorrupted sequence of 28.14dB, also shows that the CODEC itself is removing some of the noise. In some case this is due to the CODEC's having built-in filtering systems but can also be caused by other parts of a CODEC such as the quantisation stage for example.

The compressed filtered images (see Figures 9.25 and 9.27) should then decrease the effects of the CODEC resulting in an image that looks more like the original. As can be seen from all of the results that have been filtered prior to compression, the noise and CODEC artefacts are much less noticeable than had no filtering been done (see Figures 9.24b and 9.26b). The first of these (see Figures 9.25a and 9.27a) show the best filter, which uses the recursive method (see section 9.2.2) with a window size of 3. The 2D and 3D power attributes used values of 243 and 905 respectively and gives a PSNR of 34.89dB. This shows a significant drop in the visibility of noise and coding artefacts, which makes it much more pleasing to look at. However when compared to the compression results for the spatial Gaussian filtering method (see section 9.1.4), which gave a PSNR of 36.72dB for the H263 CODEC and the Foreman sequence, the morphological method is shown to perform worse. The compression results from using the Gaussian filter (see section 9.1.4) are used as a basis for comparison and are shown in Figures 9.24c and 9.26c for Frames 10 and 100 of the Foreman sequence respectively. This filter gives a PSNR, when compared to the original sequence of 36.76dB. Hence whilst these filters are the best for noise reduction, neither produce sequences that are as compressible as those produced by the Gaussian. However comparing the Gaussian with any of the morphological methods (see Figures 9.25 and 9.27), it can clearly be seen that a significant amount of noise is still visible in the Gaussian filtered sequence, which has been confirmed by showing these sequences to a few observers (see section 9.4). As was seen with the noise reduction, even a filter that is not the best, for example by using a non-recursive structure with 2D and 3D power attribute values of 584 and 4199 respectively still results in a sequence that is much better than had no processing been done. This method gives a PSNR of 32.39dB, so is not as good as the best filter combination but still shows a significantly better looking image, although some noise can be seen. The worst filter combination, which also uses the non-recursive method but with a window size of 3 and 2D and 3D power attribute values of 600 and 5857 respectively can still be seen to give an improvement, although some noise can still be seen.

a) Uncorrupted frame 10 from the Foreman sequence compressed, which gives a PSNR of 40.54dB.

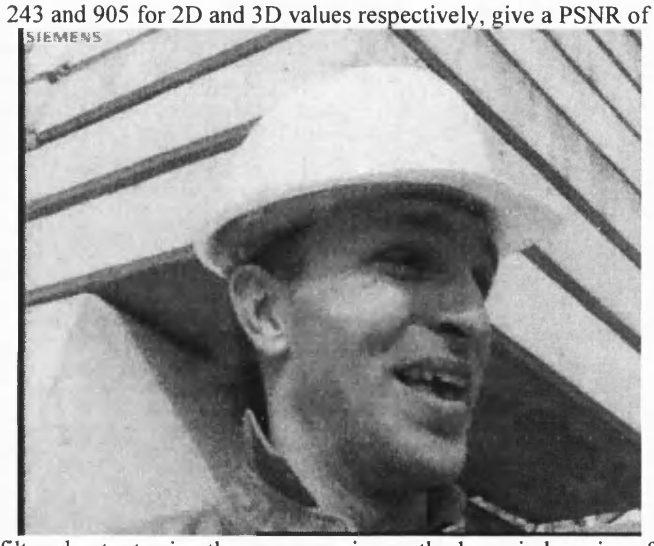
b) Frame 10 corrupted with 28dB of noise and compressed resulting in a PSNR of 30.25dB.



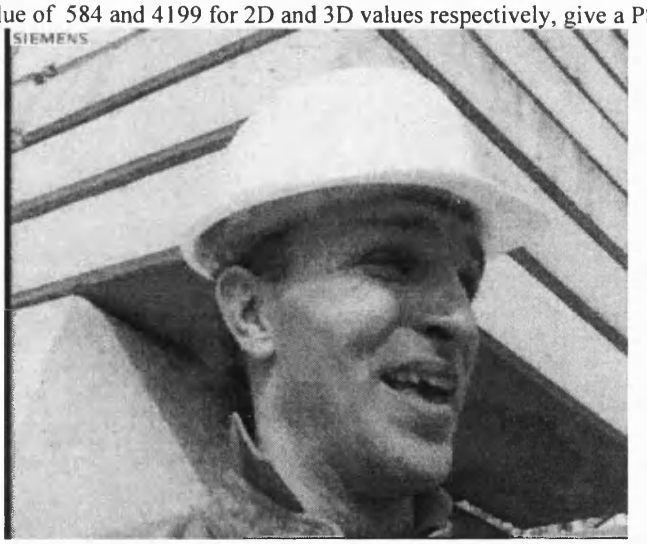
c) Frame 10 filtered and compressed with the best Gaussian filter resulting in a PSNR of 36.76dB.

Figure 9.24: The original and corrupted frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence compressed to 20:1 (0.4bpp or 990Kbps) using the H263 CODEC.

a) Best filtered output using the recursive method, a window size of 3 and the power attribute with a value of 243 and 905 for 2D and 3D values respectively, give a PSNR of 34.89dB.



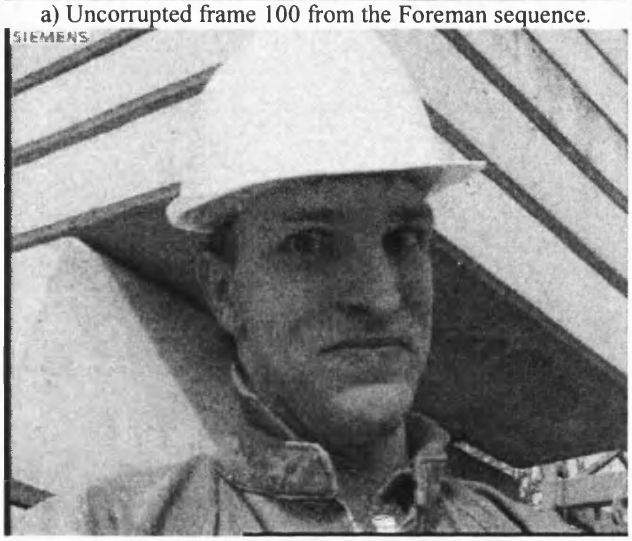
b) Intermediate filtered output using the non-recursive method , a window size of 3 and the power attribute with a value of 584 and 4199 for 2D and 3D values respectively, give a PSNR of 32.39dB.



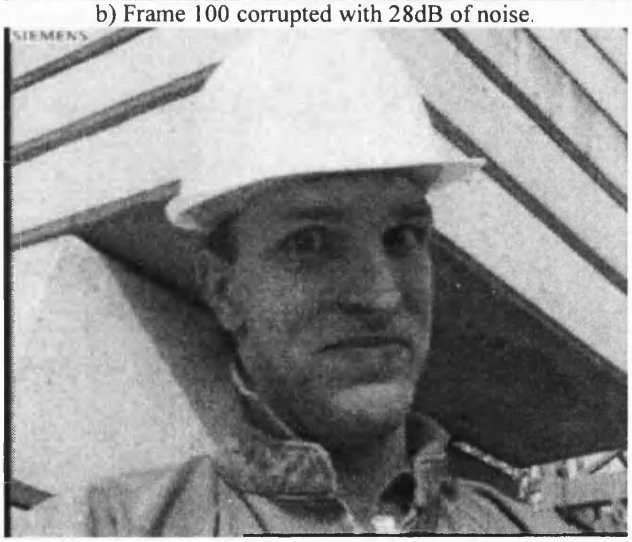
c) Worst filtered output using the non-recursive method , a window size of 7 and the power attribute with a value of 600 and 5857 for 2D and 3D values respectively, give a PSNR of 32.34dB.

Figure 9.25: Compression of the simultaneous 2D and 3D filtering using frame 10 of the 8bit greyscale, 352 x 288 Foreman sequence to a ratio of 20:1 (0.4bpp or 990Kbps).

a) Uncorrupted frame 100 from the Foreman sequence.

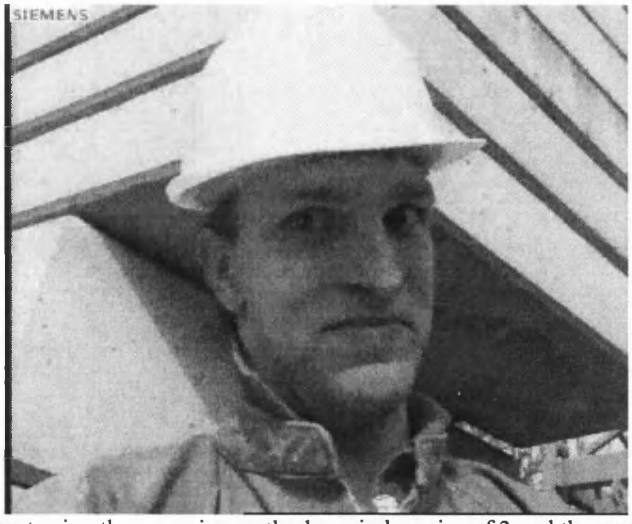


b) Frame 100 corrupted with 28dB of noise.

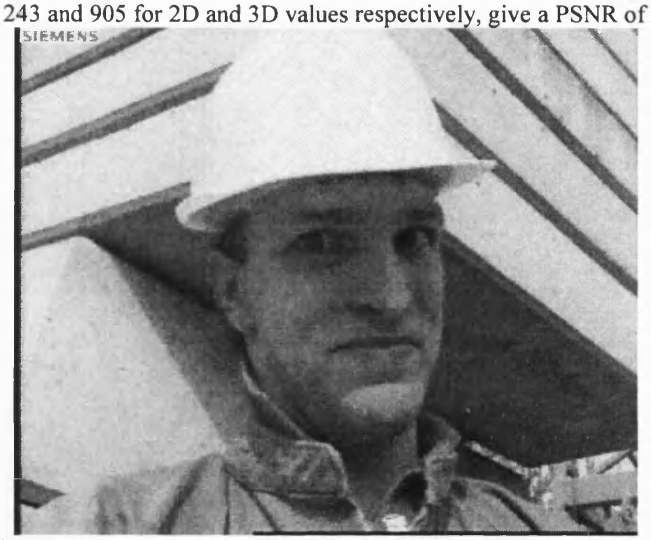


c) Frame 10 filtered and compressed with the best Gaussian filter resulting in a PSNR of 36.76dB.

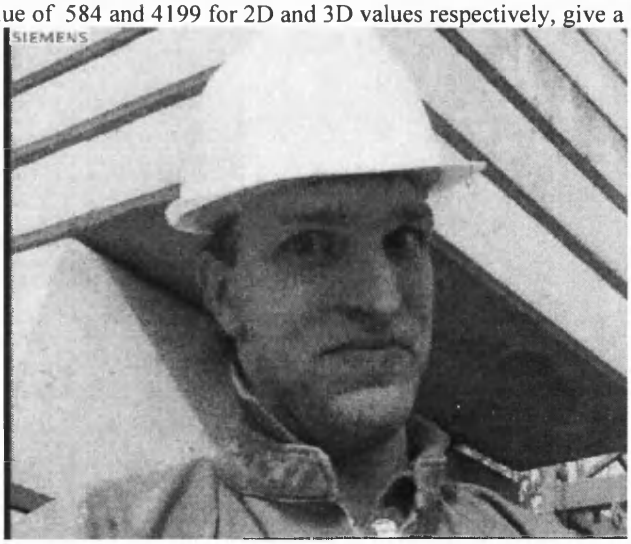
Figure 9.26: The original and corrupted frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence compressed to 20:1 (0.4bpp or 990Kbps) using the H263 CODEC.



a) Best filtered output using the recursive method, a window size of 3 and the power attribute with a value of 243 and 905 for 2D and 3D values respectively, give a PSNR of 34.89dB.



b) Intermediate filtered output using the non-recursive method , a window size of 3 and the power attribute with a value of 584 and 4199 for 2D and 3D values respectively, give a PSNR of 32.39dB.



c) Worst filtered output using the non-recursive method , a window size of 7 and the power attribute with a value of 600 and 5857 for 2D and 3D values respectively, give a PSNR of 32.34dB.

Figure 9.27: Compression of the simultaneous 2D and 3D filtering using frame 100 of the 8bit greyscale, 352 x 288 Foreman sequence to a ratio of 20:1 (0.4bpp or 990Kbps).

9.4 Psychovisual Evaluation

Four observers were asked to look at and judge the quality of the Gaussian (see sections 5.3.1.2 and 9.1.1), K-NN (see sections 5.3.3 and 9.1.2) and two best morphological filtering methods, which use the following filter combinations:

- The simultaneous 2D and 3D filtering method (see section 9.3.2) with 4nn connectivity, the AF filter structure and the power attribute and in the recursive mode, which gives the best result for the Foreman sequence,

- The combined 2D and 3D method (see section 9.3.1) using a non-recursive mode, 4nn connectivity, the AF filter structure and the power attribute, which produces the best result for the Kiel sequence.

All of the observers agreed that the K-NN method produced the worst looking results for the Kiel and Foreman sequences. This produced blurring and visually noise was still very much visible. Most of the observers agreed that the Gaussian output also showed blurring, which was noticed more on edges of objects more than anywhere else, but that this was not as severe as that produced by the K-NN filter. The two morphological filters however proved to be indistinguishable from each other. In addition the observers said they preferred these filters as the sequences were smoother, both spatially and temporally. Noise was also noted to be much less apparent on the morphological results, thus showing that the morphological filters are not only the best numerically for reducing noise but that they are also desirable for their psychovisual qualities.

Observers were also asked to evaluate a few compressed sequences. All of the observers said that the MPEG-II and H263 CODEC's produced better outputs than the BWV or MJPEG. The MJPEG CODEC was the worst due to the blocking artefact but it was also noted that comments were made about a significant amount of artefacts appearing in the BWV CODEC although not as visually annoying. For all of the CODEC's, all of the observers said that the morphologically filtered sequences appeared to be the best as they were smoother. They could not tell the difference between the two morphological filters. The Gaussian was noted to be almost as good as the morphological. It was not as good because of some noise still being visible as flicker between frames and a few observers saying that the frames had blurred in places. The K-NN filter was noted to be the worst producing flicker in some areas but more noticeable was the blurring, especially on the unseen data. However, a few comments were made that it was difficult to see differences with the MJPEG and BWV CODEC's by a couple of observers. This shows that visually the morphological filters are far superior to the K-NN and Gaussian. However it has highlighted the fact that more compression ratios should be used for evaluating the visual performance of a CODEC as at a compression ratio of 20:1, observers couldn't see much of a difference in the MJPEG CODEC. Thus at lower compression ratios, they should be able to.

9.5 Application to Unseen Data

In the real world, for example a television receiver, a sequence of images is received in a degraded state with no reference or access to the original image. Thus the sequence must be filtered using little or no prior knowledge. Like the application to unknown data for the images (see section 7.1.3), the noise variance of the sequences is measured and the filter parameters are plotted against them as shown for the Gaussian (see section 5.3.1.2) σ value and the number of neighbours to use for the K-NN filter (see section 5.3.3) in Figure 9.28. Since only two sequences are used, the LMS fir will be linear, hence only a linear fit has been used.

There are two morphological filters that perform better than the rest, the first uses the simultaneous 2D and 3D filtering with 4nn connectivity, the AF filter structure and the power attribute and in the recursive mode, which gives the best result for the Foreman sequence. This gives a PSNR of 33.20dB, an improvement of 5.06dB. The second uses a non-recursive mode, 4nn connectivity, the AF filter structure and the power attribute but using the combined 2D and 3D method, which produces the best result for the Kiel sequence. This gives a PSNR of 30.28dB, an increase of 2.16dB. Hence only these two morphological filters are evaluated using unknown data. All of the filters show an increase in the filter values as the noise variance increases with the exception that the mask size for the Gaussian method is 3 x 3 for both sequences, hence this is then assumed to be the best size to use In addition both of the best morphological filters use a window size of 3, which is also assumed to be the best size to use. Linear LMS fits are made to the data so that the noise variance of a sequence can be estimated and used to evaluate the filter values to be used. For the Gaussian σ, the equation evaluated to:

$$\sigma = 0.0209x - 1.4214 \tag{9.2}$$

where x is the estimated noise variance. Similarly, the K-NN fit for finding the best number of neighbours to use evaluates to:

$$K = 2.0919x - 183.12 \tag{9.3}$$

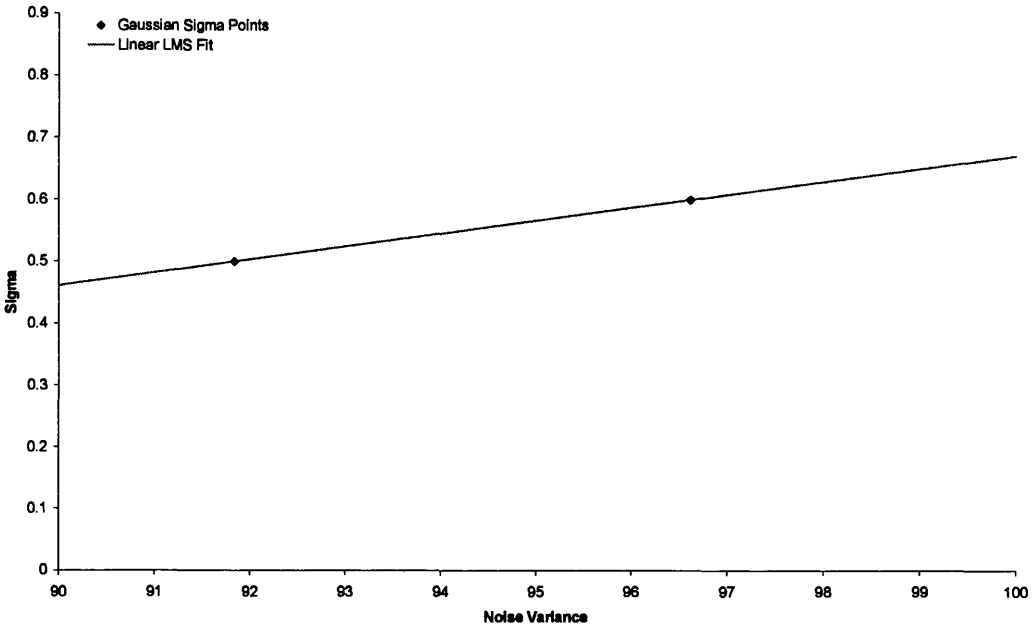
These two best morphological filters are evaluated in the same way, but like the Gaussian have two variables, one for the 2D filter value and one for the 3D filter value. Equations 9.4 and 9.5 give the formulas for the filter values using the simultaneous method and equations 9.6 and 9.7 give the formulas using the combined method.

$$Power_{2D} = 20.501x - 1737.8 \tag{9.4}$$

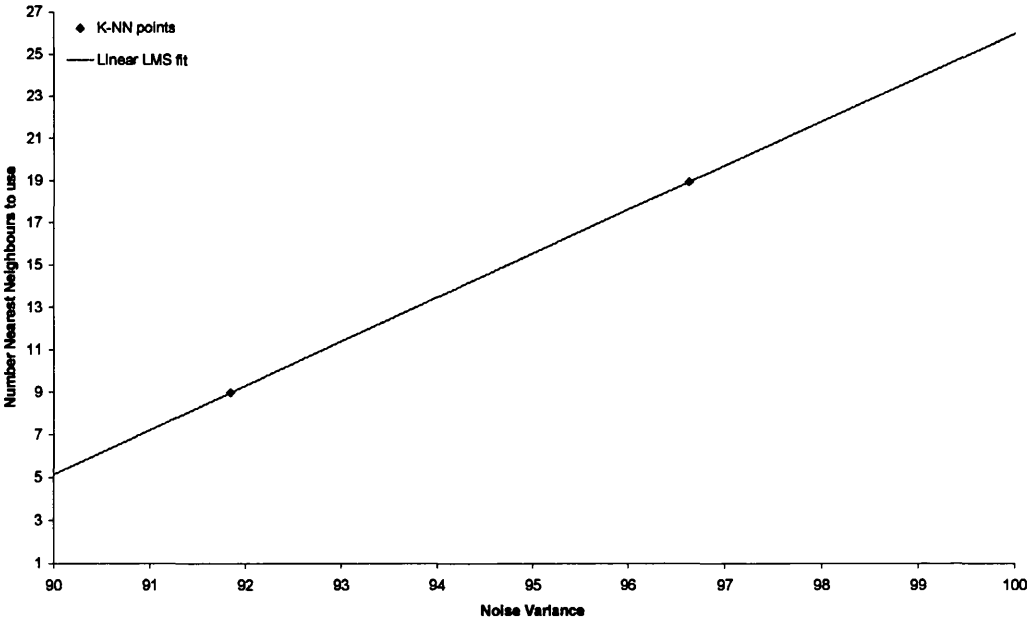
$$Power_{3D} = 78.447x - 6674.6 \tag{9.5}$$

$$Power_{2D} = 176.77x - 15494 \tag{9.6}$$

$$Power_{3D} = 19.873x - 1678.2 \tag{9.7}$$



a) Plot of the best Gaussian filter points for σ and a Linear LMS line fitted to the data.



b) Plot of the best K-NN filter points for K and a Linear LMS line fitted to the data.

Figure 9.28: Noise variance against filter values.

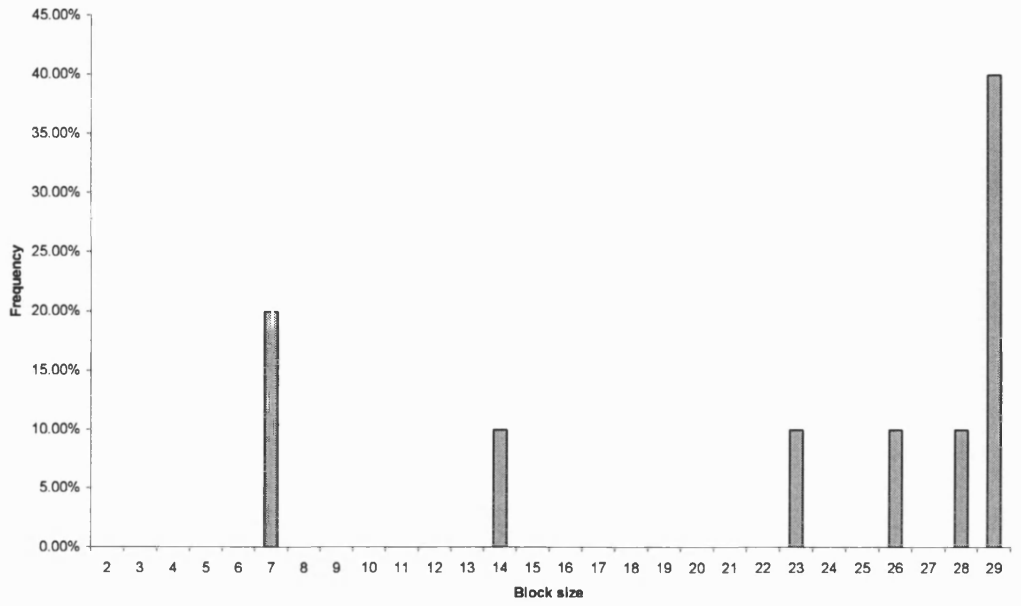
9.5.1 Estimating the Noise Variance

Like the spatial method (see section 7.1.3), in order to estimate the filter values to use, an estimate of the noise variance needs to be made. The same technique as the spatial is used, that is that the image is split into blocks, the variance of each block is calculated and the average of a certain percentage of the smallest variances is used as the estimate of the true noise variance. There are several ways in which this can be extended to sequences. One method is to calculate the noise variance for each frame and then average them all together. However, the method used calculates the variances for all of the blocks over the entire sequence.

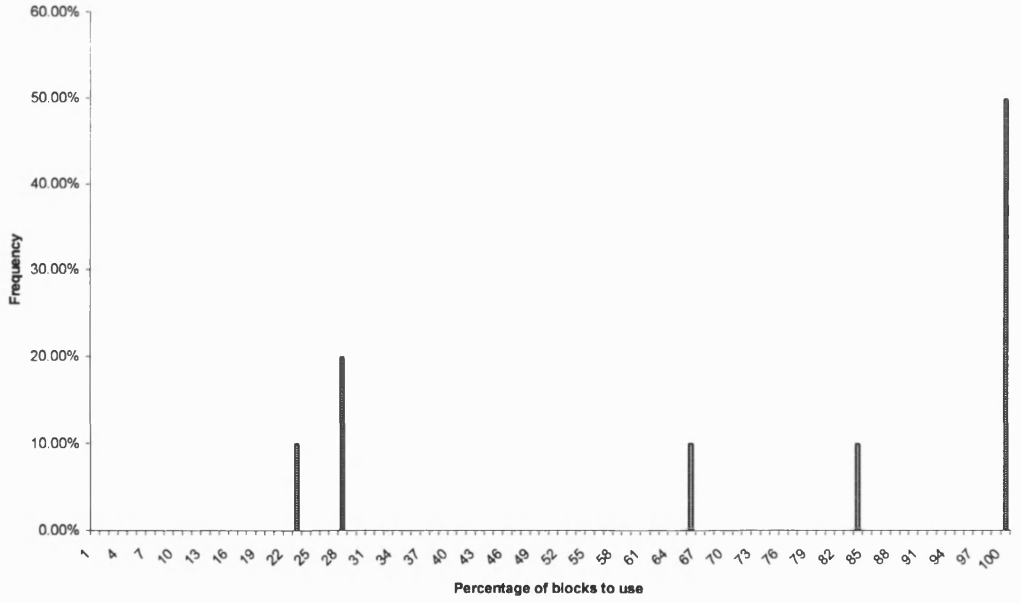
There are two variables that can be changed with this method, the block size and the percentage of blocks to use in order to calculate the noise variance. Over 10 sequences were corrupted with 28dB of noise and the best values for block size and percentage to use are found that produce the closest noise variance estimation to the true noise variance. Figure 9.29 shows the frequency the block sizes and percentage of block sizes to use occur to use to find the closest match for noise variance estimation. This shows that a block size of 29 x 29 is the most frequently used block size, that is it is used to obtain the closest match 40% of the time, where as the second graph shows that 100%, all of the blocks are used to obtain the best estimation for 50% of the results. However, a block size of 29 x 29 may not necessarily use all of the blocks, hence two further experiments are conducted. The first (see Figure 9.30a) uses 100% of the blocks to calculate the noise variance but varies the block size to find the best. This shows that a block size of 29 x 29 occurs more often, 50% of the time, than any other. The second test uses a block size of 29 x 29, which was indicated to be the best block size from the previous experiments, and varied the percentage of blocks to use in order to estimate the noise variance. This showed that the most frequent outcome, 50% of the time, used 100% of the blocks. This indicated that using all of the blocks and a block size of 29 x 29 will provide the best overall estimation of the noise variance.

9.5.2 Filtering Unseen Data

Like the spatial method (see section 7.1.3.2), an uncorrupted sequence is chosen, Stefan, and corrupted AWGN (see section 5.1.1) resulting in a PSNR of 28.16dB. The original sequence is not used in the filtering process, but is used after filtering to provide an evaluation of the performance of the system. The first 50 frames of the Stefan sequence is used, which a size of 352 x 288 pixels and is shown at 25fps. Only the Y channel is used making the sequence 8bit greyscale. Using the noise estimation method (see section 9.5.1) to evaluate the variance of the noise gives a value of 93.77. This is then used to calculate the filter values by replacing x in equations 9.2 – 9.7, which gives the filter values as shown in Table 9.20, which are subsequently used to filter the sequence with. The original sequence is then used to evaluate the performance of the system in terms of PSNR. The K-NN filter is shown to be the worst performing, as it was when the best filter values were used (see section 9.1.2). Morphological filtering proves to outperform the Gaussian, with the simultaneous method performing the best with a PSNR of 30.45dB, which is and improvement of 2.29dB.

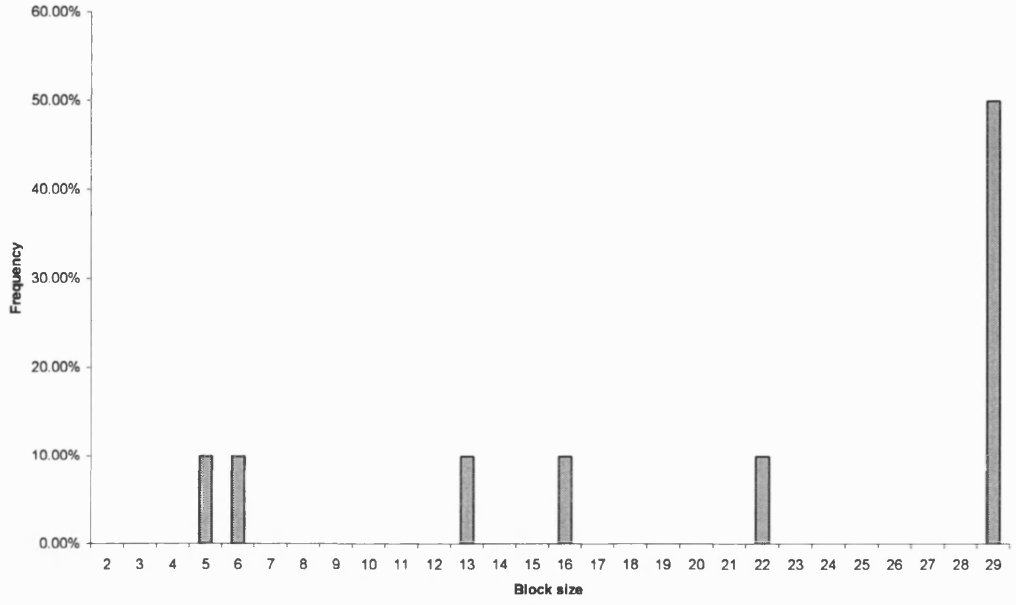


a) Frequency of the best block sizes to use.

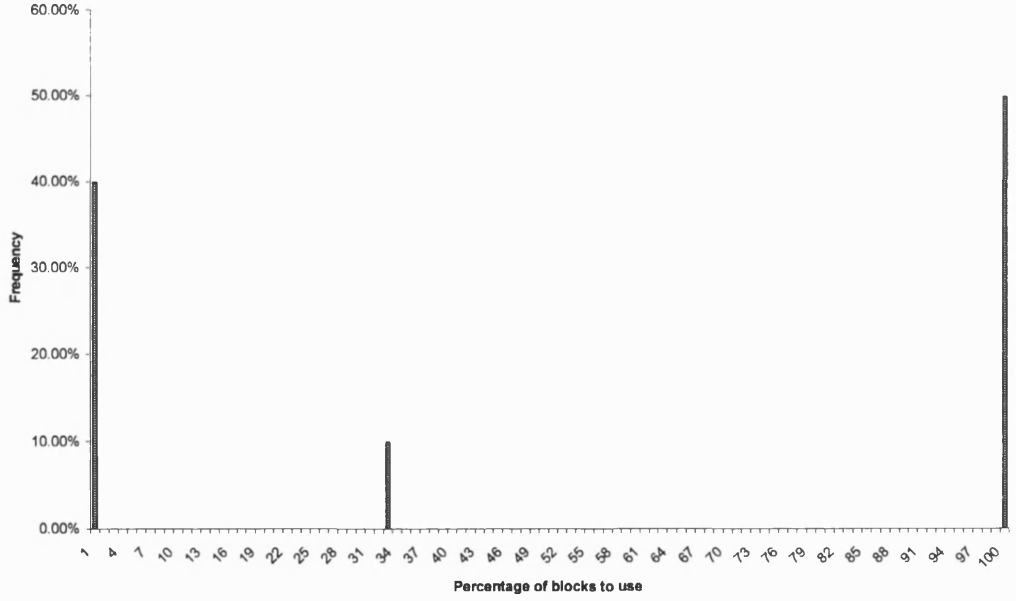


b) Frequency of the best percentage of blocks to use.

Figure 9.29: Frequency of the best block size and percentages of blocks to use.



a) Frequency of the best block sizes to use using 100% of the blocks to calculate the noise variance.



b) Frequency of the percentage of blocks to use when calculating the noise variance with a 29 x 29 block size.

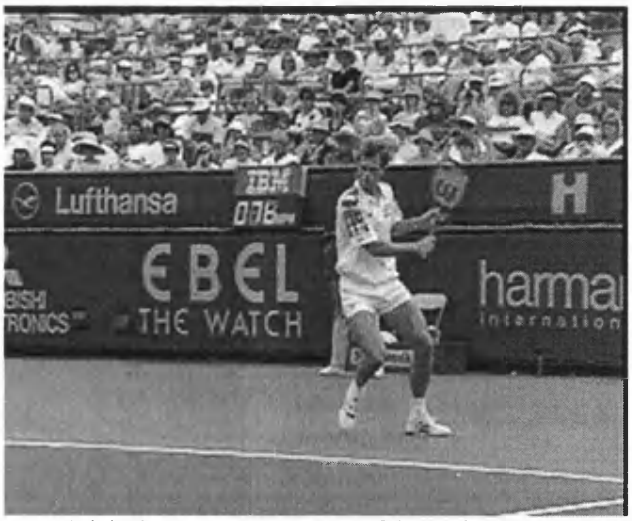
Figure 9.30: Frequency of the best block size and percentages of blocks to use.

This has shown that all of the filters can be adapted to allow for blind filtering of a sequence and that the morphological filtering methods work better than the Gaussian and K-NN. However, the performance gain of the morphological over the Gaussian is relatively small. Further refinements, for example using many more sequences and noise levels to tune the system may increase this and provide a much more reliable and accurate estimation of the filter parameters.

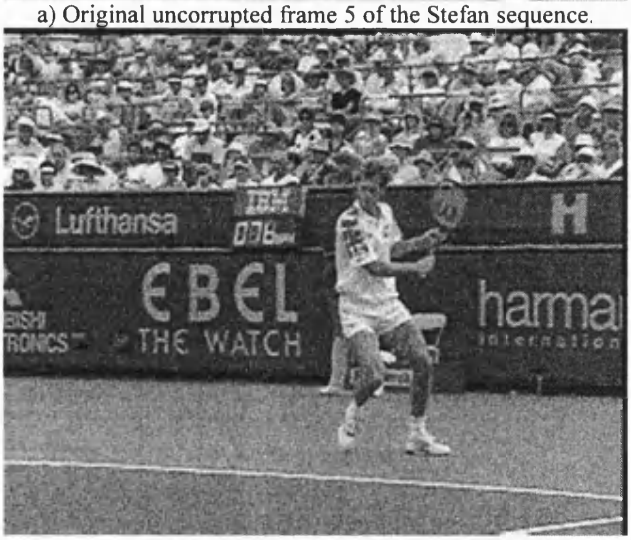
Filter	Filter values		PSNR (dB) of filtered sequence
	2D	3D	
Gaussian, using a 3 x 3 mask	0.5		30.28
K-NN	13		28.60
Morphological – Simultaneous Using 4nn connectivity, Power attribute and AF filter structure with the recursive method, using a window size of 3	185	682	30.45
Morphological – Combined Using 4nn connectivity, Power attribute and AF filter structure with the non-recursive method, using a window size of 3	1082	185	30.39

Table 9.20: Estimated filter values to use and the resultant PSNR after filtering.

Figures 9.31 – 9.34 show example frames, frames 5 and 45, from the Stefan sequence. Figure 9.31 shows the original and corrupted frame 5 of the 352 x 288, 8bit greyscale Stefan sequence and Figure 9.33 shows the same for frame 45. In addition the Gaussian filtered frame is shown using the parameters estimated (see Table 9.20). Whilst on a frame by frame basis the Gaussian appears to remove a great deal of noise, when viewed as a video, noise is still very much visible as pixels flicker between frames. The K-NN filter, see Figure 9.32a, shows the frame being blurred, both spatially and temporally. The noise is more apparent than it was in the output of the Gaussian filter. The two morphological filters, see Figures 9.32b, 9.32c, 9.34b and 9.34c all look very similar. However, it was noticed that using the simultaneous method, some temporal blurring occurred, although only slightly and only in a few places. Since this was not observed in the training sequences, this could be caused by the estimation of the attributes not being accurate enough and hence more training of the system is required. It is however noted that the morphological methods produce a much smoother sequence, both spatially and temporally than either the Gaussian or K-NN filtering methods.



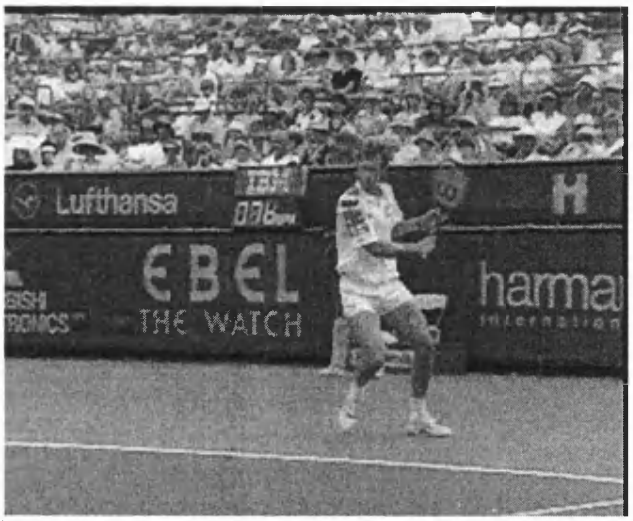
a) Original uncorrupted frame 5 of the Stefan sequence.



b) Corrupted frame 5 with 28dB of AWGN.

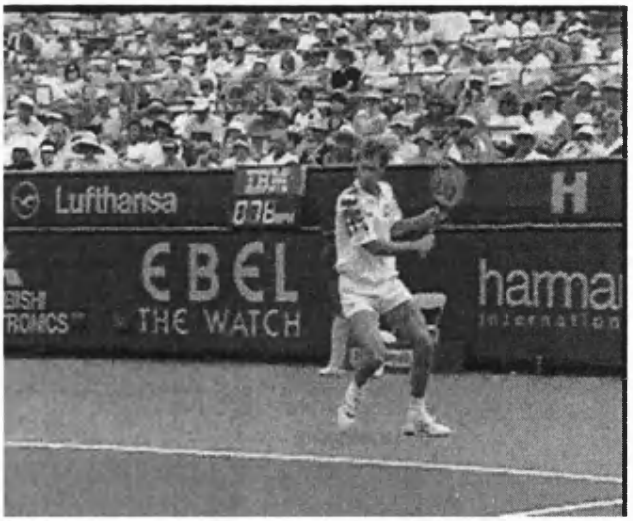
c) Frame 5 after the sequence has been filtered using the Gaussian method with a 3 x 3 mask and $\sigma = 0.5$. This gives a PSNR of 30.28dB.

Figure 9.31: Original, corrupted and Gaussian filtered frame 5 of the 352 x 288 8bit greyscale Stefan sequence.



a) Frame 5 after the sequence has been filtered using the K-NN method with K=13, which gives a PSNR of 28.60dB.

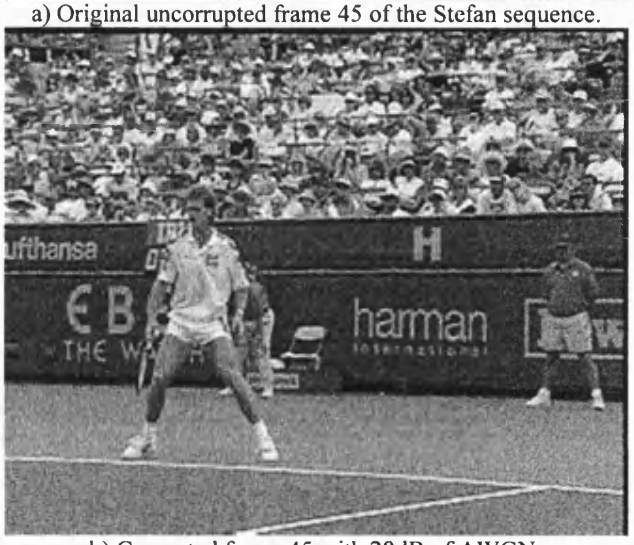
b) Frame 5 after the sequence has been filtered using the simultaneous morphological method giving a PSNR of 30.45dB.



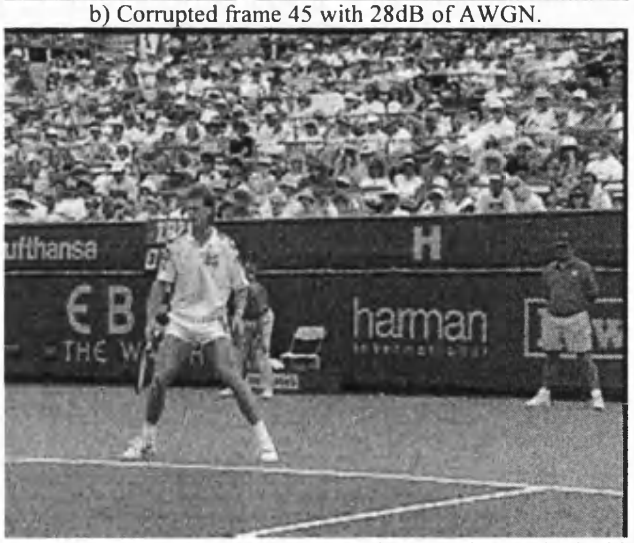
c) Frame 5 after the sequence has been filtered using the combined morphological filtering method, which results in PSNR of 30.39dB.

Figure 9.32: Frame 5 of the 352 x 288 8bit greyscale Stefan sequence filtered using the K-NN and morphological methods.

a) Original uncorrupted frame 45 of the Stefan sequence.



b) Corrupted frame 45 with 28dB of AWGN.



c) Frame 45 after the sequence has been filtered using the Gaussian method with a 3 x 3 mask and $\sigma = 0.5$. This gives a PSNR of 30.28dB.

Figure 9.33: Original, corrupted and Gaussian filtered frame 45 of the 352 x 288 8bit greyscale Stefan sequence.

a) Frame 45 after the sequence has been filtered using the K-NN method with K=13, which gives a PSNR of 28.60dB.

b) Frame 45 after the sequence has been filtered using the simultaneous morphological method giving a PSNR of 30.45dB.

c) Frame 45 after the sequence has been filtered using the combined morphological filtering method, which results in PSNR of 30.39dB.

Figure 9.34: Frame 45 of the 352 x 288 8bit greyscale Stefan sequence filtered using the K-NN and morphological methods.

9.5.3 Compression Results

Previous results from this chapter have shown that filtering improves the compressibility of a sequence when the best filter is available. However since the filter parameters have been estimated and therefore may not be the best, this may also mean a gain in the quality of the compressed sequence is achieved. Thus the filtered sequence is compressed and decompressed using the four CODEC's, MJPEG, MPEG-II, BWV and H263. The PSNR is then measured against the original uncorrupted sequence, the results of which are given in Table 9.21. This shows that all of the filters produce a better output than had the sequence not been filtered. For example, using the H263 CODEC with the noisy sequence, a PSNR of 29.02dB is achieved. Using the combined morphological approach, a PSNR of 32.50dB is achieved. In addition the K-NN filter is shown to produce results the same as, or for most of them worse than the other filters with the Gaussian producing the best results. Of the two morphological methods, the combined method produced the best results which is the opposite outcome of the noise filtering alone where the simultaneous method worked best. Gaussian filtering results are very close to the Combined morphological method, further refinement and tuning of the filter estimation may decrease this difference.

Filter	CODEC	PSNR (dB)
Unfiltered, original sequence	MJPEG	27.44
	MPEG-II	30.46
	BWV	28.63
	H263	34.84
Unfiltered, noisy sequence	MJPEG	25.80
	MPEG-II	26.58
	BWV	26.36
	H263	29.02
Gaussian	MJPEG	**28.34**
	MPEG-II	**29.49**
	BWV	**28.96**
	H263	**32.50**
K-NN	MJPEG	27.78
	MPEG-II	29.04
	BWV	28.69
	H263	31.63
Morphological – Simultaneous Using 4nn connectivity, Power attribute and AF filter structure with the recursive method, using a window size of 3	MJPEG	27.97
	MPEG-II	29.27
	BWV	28.75
	H263	32.33
Morphological – Combined Using 4nn connectivity, Power attribute and AF filter structure with the non-recursive method, using a window size of 3	MJPEG	28.05
	MPEG-II	29.40
	BWV	28.86
	H263	**32.50**

Table 9.21: PSNR of the Stefan sequence after compression to a ratio of 20:1 (0.4bpp), with the best output for each CODEC shaded, not including the unfiltered sequences.

9.6 Conclusion

This chapter has shown how mathematical morphology can be used to filter image sequences. Several filtering methodologies have been developed and evaluated. The spatial only method is the least complex. Several efficient methods (see chapter 6), such as Wilkinson's method (see section 6.1.5) and the Max-tree (see section 6.1.4) can be used. However, the goal of this chapter was to investigate different filtering methods and not the efficient implementation of them.

The results of the noise reduction have all show an improvement on the PSNR regardless of the filtering method used. Gaussian filtering (see section 9.1.1) was shown to work well giving a 1.62dB and 3.97dB increase in PSNR for the Kiel and Foreman sequences respectively, which the morphological methods have exceeded. However observers could see some blurring in the filtered output and in some cases, flickering between frames, which is caused by the difference between frames, for example the motion of a tennis ball. The K-NN filter (see section 9.1.2) was also shown to be effective at noise reduction giving an increase of 0.98dB and 3.3dB for the Kiel and Foreman sequences respectively. This is not as good as the Gaussian but is much more efficient to implement. Observers though thought that sequences filtered using this method did not look as good as those filtered using the Gaussian method. The frames appeared more blurred both spatially and temporally.

The morphological filtering methods though showed a much higher gain in PSNR, 2.14dB and 4.92dB for the Kiel and Foreman sequences respectively. These are the best results from the spatial morphological filters and both used the power attribute (see section 6.2.2.4), 4nn connectivity and the AF filter structure (see section 4.6.2), thus showing that the power attribute is a better method to use than the area attribute. In addition, the AF filter structure is faster than the ASF (see section 4.6.2 and chapter 6), as is the 4nn connectivity when compared to the 8nn connectivity (see chapter 6). Hence this results in the fastest combination of the morphological filter. Observers also said that they preferred this output to the K-NN filtered sequences, but there was an even split (2 observers for each) when compared to the Gaussian. None of the observers could see that the morphological creates flat areas although they could see that it was smoother in appearance.

Despite the apparent success of the morphological approach, it can still produce sequences that flicker. Hence spatio-temporal methods were developed in several different ways. The first implemented a block based preprocessing system as was shown in section 9.2. The results of this, which was an increase of 1.98dB and 4.37dB for the Kiel and Foreman sequences respectively, although not as high as the spatial only method, still performed better than both the Gaussian and K-NN methods. In addition since this is a spatio-temporal system, there is less flicker between frames, although there can still be flicker between the block. Since only spatio-temporal points are taken, there is still a large amount of spatial extrema that are left un-processed. Two further improvements were made to this system, the first of which processed a block as before, but only save the centre frame. The block was then slid along by one pixel and the process repeated. The second method used the same process, but kept the previous filtered data when the window is moved. Results from these two methods (see section 9.2.2), which was an increase in PSNR of 2.03dB and 4.6dB for the Kiel and Foreman

sequences respectively, showed that keeping the previous results produced the best results in conjunction with the AF filter structure and a window size of 3 with the power attribute. This is an improvement upon the block method but still is not as good as the basic spatial method.

To obtain the best of both worlds, the spatial and spatio-temporal, two hybrid methods were developed that combine spatial and spatio-temporal processing. The first method described in section 9.3.1 simply processes the sequences spatially and then spatio-temporal using one of the 3 spatio-temporal filtering methods. The best results showed an improvement of 2.16dB and 5.00dB for the Kiel and Foreman sequences respectively. This beat the spatial only method by a very small amount, however it does reduce the flickering and removes more of the noise. This method again shows the power attribute and the AF filter structure performing the best in addition to the overlapping, recursive spatio temporal method. Since this method is sequential, it increases the processing time, which is countered by the fact that the spatio-temporal attributes are much smaller than they would have been if not preceded by a spatial filter and that spatial filtering reduces the number of spatio-temporal regions. The second method processes the spatial and spatio-temporal data simultaneously (see section 9.3.2). The results shown an improvement of 2.13dB and 5.06dB for the Kiel and Foreman sequences respectively. Thus both are better than the spatial only results but the Kiel is worse than using the sequential method of filtering spatially and then spatio-temporally. However the Foreman results do show the best overall results. Observers could not tell the difference between the two techniques and thus is hard to pick the best method, although the simultaneous method is slightly more complex to implement. To pick the best method, further investigations into efficient implementations and psychovisual evaluations on the filtered data would need to be carried out of a large number of sequences and observers. This has however shown two successful methods for noise reduction using new spatio-temporal morphological filtering methods.

The compression results for the spatial methods however show that the Gaussian and K-NN filtering methods produced a more compressible sequence. For example compressing the corrupted Foreman sequence to a ratio of 20:1 (0.4bpp or 990Kbps), which had a PSNR of 28.14dB, produces an output PSNR of 32.16dB when used with the H263 CODEC (see Table 9.3). Using the best Gaussian filter prior to compression results in a PSNR at the output of the CODEC of 36.76dB. The morphological spatial filter manages only 34.71dB. This suggests that either the Gaussian is filtering better or is simply filtering more. To determine which it was, four observers were asked to look at the compressed sequences (see section 9.4). Using the MJPEG and BWV CODEC's, all of the observers found it difficult to see differences due to the artefacts introduced by the CODEC's. However for both the MPEG-II and H263 CODEC's, all of the observers still said that the morphologically filtered sequences looked the best, generally because they looked smoother, less blurred and couldn't see any distortions as the sequence was played, especially on the Stefan sequence. Hence although the PSNR shows the Gaussian performing best in conjunction with the CODEC's, simple psychovisual tests show that this is not the case visually. In addition, since the filters have been tuned for noise removal, further gains may be possible by retraining the system but using the CODEC outputs to train the system instead of the filter outputs. This may increase the compressibility of the sequences, but would have the disadvantage that it would be tuned for a particular CODEC, whereas the current

system can be used in conjunction with any CODEC. However all of the compressed sequences show an improvement on the PSNR of the un-filtered sequences after being compressed. Thus all of the methods have fulfilled the original goal of reducing the noise and improving compressibility. To this end, the spatial method would then be the best method to use in a real world system where a real-time system is required for the simple reason that it can be implemented very efficiently (see chapter 6). However, further investigation into spatio-temporal processing could investigate and produce more efficient implementation methods that would allow a real-time system to be developed.

Throughout this chapter and chapter 7, the power attribute has been shown to perform better than the area attribute for the majority of cases. However, unlike the spatial methods (see chapter 7), the spatio-temporal methods have shown that the AF filter structure generally outperforms the ASF filter structure for noise reduction regardless of the image sequence used. This may be because of the difference in the image size, the content of the frames or that sequences have different properties to images for morphological filtering.

The final development of these filtering methods was to apply them to unseen data. Hence section 9.5 shows the application to unseen data. A sequence, Stefan, was corrupted with noise and the noisy sequence then filtered without using the original sequence. This showed that the variance of the noise could be estimated and used to evaluate the filter parameters to use. All of the filters showed an increase in performance, although the morphological filter using the simultaneous spatial and spatio-temporal filtering (see section 9.3.2) showed the highest improvement of 2.29dB. However the Gaussian filter proved to produce the best compressible output as it did before. Observers however still said that they preferred the morphological output as it showed less flicker between frames, was smoother and did not blur the frames like the K-NN and Gaussian filters did. A difference between the two morphological filters, the simultaneous and combined (see section 9.3.1) could still not be seen. Hence this shows that attribute values can be estimated and used to filter any sequence, although more training may result in a better estimate of the attribute values. In addition more noise levels could be used, thus allowing the filters to be used not only on any sequence but also with any amount of noise.

This chapter has shown several new image sequence noise reduction methods using mathematical morphology. All of the methods have been shown to be able to reduce the noise and improve the compressibility of image sequences. In addition the noise reduction has been shown to perform better than two other currently used techniques, the Gaussian filter (see sections 5.3.1.2 and 9.1.1) and the K-NN filter (see sections 5.3.3 and 9.1.2) and that psychovisually the morphological sequences produce a better output from a CODEC than other methods. Application to previously unseen data has been shown to work, although further refinement could be used to increase the accuracy and range of noise levels that could be filtered.

10 Conclusions

In image compression, the ability to compress data as efficiently as possible is a key point. Regardless of the technology used to obtain material, the images will inevitably contain some degree of noise and degradation. This can have a significant impact on the performance of an image compression system due to the amount of extra high frequency information present. This thesis has shown the basics of digital video systems and several methods, including mathematical morphology for improving the compressibility of an image compression system. These methods include both the reduction of noise and psycho-visually lossless image simplification.

Digital and analogue video systems were introduced in chapter 2 and the history and development of the television was shown. A brief overview of digital video was covered which showed why digital is considered better than analogue and why compression is required for digital transmission. To show how this works the MPEG-II CODEC was introduced, from which most CODEC's work in a similar way. Digital systems still have some disadvantages compared to analogue systems, some of which include artefacts, degradation in performance due to noise, delays in coding, etc.

The HVS is described in chapter 3, which showed how it works and several ways to measure image quality. The two methods, objective and subjective, have been clearly shown. Objective measurement is by far the most widely used measurement as it is easy to implement, produces consistent values and is cost effective. However, this can be flawed as was shown in section 3.2.1, in that an objective measurement can give a value that indicates an image that will show little degradation or artefacts. In reality it may look worse than an image with a poor objective measurement. Thus the subjective measurement was introduced. This allows observers to judge the quality of images. Hence subjective measurements allow a system to be tuned to best approximate the properties of the HVS. This allows more filtering in certain areas, for example flat areas, and thus creates less data for compression. However, subjective measurements do have several drawbacks. Firstly observers need to be found to take part in the experiments. These need to be people with little knowledge of image processing, so that they do not know what too look for. In addition the same observers should not be used in multiple sittings. In addition, in a commercial environment, this often results in the observers needing to be paid and hence this method can be expensive. Finally, even if the same experiments are run again, there is little chance of obtaining the same results. This thesis has used both methods with the objective measurement being used to measure the performance of the noise reduction methods and the subjective measurement used to implement a psychovisually lossless image simplification system.

The idea of mathematical morphology was introduced in chapter **4**, which described and explained the basic operations, Dilation and Erosion. This has then been expanded on to derive more complex mathematical morphology operators such as granulometries, Open-Close filters, AM and attribute morphology filters. Binary and greyscale operators have been shown working in 1, 2 and 3

Dimensions. This thesis has used only the luminance channels of images. However mathematical morphology can be used for colour processing. The method of how this is applied is still of some debate. For example the channels could be treated independently of each other or the channels could be made dependant on each other, for example by using vector operators.

Noise was introduced in chapter 5. Several noise models have been described, including Gaussian noise, which is considered a worst-case approximation. A literature review has also been shown describing how other researchers have tackled the problem of noise reduction. Most of these tend to use impulse noise and median filters although several attempts at using them for Gaussian noise removal have been made. Mathematical morphology systems for image noise reduction were reviewed and the potential for improvement highlighted. Most systems reviewed have been designed to work with one specific CODEC. This thesis has developed a preprocessing system that will work with any CODEC. This does however have the disadvantage that the best performance may not be obtained but does allow more flexibility in the choice of CODEC. Thus the latest CODEC's can easily be used with the system developed within this thesis with little or no modifications.

The development and implementation of AM and attribute morphology methods was shown in chapter 6. The simplest method, the pixel-queue algorithm has been shown which uses morphological reconstruction. More efficient methods have also been looked at, the max-tree and Wilkinson's algorithm. This has then been applied to attribute morphology and a new attribute based on the power removed was developed. This new attribute is used throughout the remainder of the thesis and has proven to be better at noise reduction than the area, contrast or volume constraints. Two filter structures have also been discussed in section 4.6.2, the AF and the ASF. Whilst both can use the efficient implementation methods reviewed, when the ASF is used the time required to execute the filtering rises quickly as the attribute size increases. Hence to overcome this, a new efficient ASF implementation has been developed based on the pixel-queue method. This has been shown to have an almost constant processing time regardless of the attribute size. The only factor that affects the processing time is the image content. Execution times of all four methods for the ASF were evaluated using both 4nn and 8nn connectivity.

Two new methods for image preprocessing were developed in chapter 7. One based on a psychovisually lossless measurement. This system has used the subjective measurement system to obtain an attribute value for the area and power attributes using the AF and ASF filters and 4nn and 8nn connectivity. This has been shown to be visually lossless and further psycho visual evaluations carried out to confirm that the JPEG and JPEG 2000 CODEC do gain in performance from this processing. This has shown that mathematical morphology attributes can be psychovisually tuned and that mathematical morphology can be used to give a significant gain in the quality of a compressed image. The second method concentrated on reducing the noise, Gaussian, from an image. Several images and amounts of noise were used to evaluate the performance of the filter. In addition to the area and power attributes, the contrast and volume attributes have also been evaluated. Again the power attribute was shown to give the overall best results. This also showed that the attribute values used are linked to both the amount of noise used and the image contents. Thus the best attribute value

for one image is not necessarily the best for another image. If the amount of noise affecting an image is known, or a close guess is available, then the filtering can be accomplished by using a measure, the mean for example, of the optimum values found for the test images at that level. The amount of noise could be estimated using blind estimation. An appropriate attribute and value could then be selected based on the results obtained. In addition, the AF and 4nn connectivity appear to give the best performance, which gives the advantage that Wilkinson's method can be used to increase the speed of the software.

The preprocessing methods shown in chapter 7 are used to develop a several spatio-temporal noise-reduction systems in chapter 8. These form novel filter structures that have not been reported in any literature, which allow a significant gain in the noise reduction and compressibility of an image sequence. There are four primary types of filter developed for mathematical morphology preprocessing, the first method being the only method commonly reported and used in the literature, is the spatial filter, which is used for a comparison. The second filter uses a spatio-temporal only filtering method whilst the third method combines this method with the spatial in a sequential filter. That is the image sequence is first filtered spatially and then spatio-temporally. The final filter takes this last method a step further and performs the spatial and the spatio-temporal filtering simultaneously. The power attribute demonstraights a significant improvement upon the area attribute. Evaluation of this system has shown that noise can be reduced relatively easily. The 4nn and AF filter structure have consistently produced better results than any other combination, especially when the power attribute is used. This has shown the potential for mathematical morphology in digital image and video preprocessing for both psychovisually lossless preprocessing and noise reduction.

Although this thesis has obtained the goal of reducing the effects of noise on a CODEC and increasing the CODEC's performance by image simplification, there are several possible future areas of research and development that could be looked at to further increase the performance of the image and sequence processing systems:

- For ease of implementation and development, the pixel-queue algorithm has been used for the preprocessing system. Thus future work could include developing a system using the max-tree or Wilkinson's method in order to increase the speed of the system, especially for the spatio-temporal filtering and filtering using the ASF structure.

- All of the systems developed for this thesis have been specifically designed to run in software. However most real-world preprocessing systems use dedicated hardware, which are inevitably faster than software solutions. Thus future work could also consider hardware implementations on a Field Programmable Gate Array (FPGA) for example.

- The spatio-temporal filtering has only been applied to noise reduction due to both time and resources. Hence future work may also include psychovisually evaluating the spatio-

temporal methods in a similar way to the spatial methods of section 7.2, which should give a significant increase in the compressibility of a video sequence.

- Currently, only the luminance channel is processed. Hence it would be helpful to look into how colour processing should be done. For example, the chrominance may need to be dependant on how the luminance channel is processed or it may be satisfactory to leave them independent of each other. This may depend upon the colour model used. In addition this will need to be visually tested, as the HVS is less sensitive to colour information. Hence it maybe possible to filter the chrominance channels more heavily.

- The system is currently CODEC independent, and thus whilst noise reduction can be effective, the gains seen by a CODEC are lower than had other filters been used. Hence future work may also include tuning the filters developed based on the CODEC outputs rather than filtering specifically for noise reduction. This would allow the optimum amount of processing to be done to give the best increase for the CODEC output.

11 Acknowledgements

This project would not have been possible without help and support from various people. Acknowledgements are made to my supervisor *Dr Adrian Evans* who has given me invaluable help, guidance, support and encouragement over the course of this PhD. *Professor Don Monro* has provided me with access to many of the resources (laboratories, plasma screens, offices, computers, etc.) that have been used to carry out this research, without which little research would have been accomplished. *Professor Arthur Mason* from Tandberg Television for his continued support of this project. Much of this research would not have been possible if not for the help of many other staff members including; *Rosemary Ainsworth*, *Dorcas Mumford*, *Emma Bryant*, *Graeme Everton* and *Dr Owen Holbrook* for help with the administration, finance and computer support. Finally, thanks to all members of the Signal and Image Processing Group (SIPG) of the University of Bath for the help, support and encouragement they have given to me.

12Appendix A - Set Theory

This chapter contains a basic introduction to set theory, which mathematical morphology uses as a foundation. For a more detailed analysis, the reader is referred to [43], [44], [145], [146].

12.1 Sets

A set can be defined as a group of elements or objects. In general, a set is defined by the use of an upper case letter, 'A', and an element is expressed as a lower case letter 'x'. For example, a set A could contain all even numbers from 1 to 10 as given by equation 12.1. There are many different ways in which the same set can be defined. For example, the set above could also be defined as shown in equation 12.2.

$$A = \{2,4,6,8,10\} \tag{12.1}$$

$$A = \{x : x \text{ is positive, x is even, x is a whole number and } x < 12\} \tag{12.2}$$

This reads; A is equal to x such that x is positive, x is an even number and x is less than 12. Sometimes, sets will contain no data at all. Such sets are called '**empty sets**' and are defined as:

$$A = \{\ \} \text{ Or more commonly as, } A = \phi \tag{12.3}$$

12.1.1 Venn Diagrams

Venn diagrams give a visual aid in which set theory can be viewed with ease. Sets are usually depicted as circles while the space in which the sets are located is usually drawn as a rectangle. For example, given two sets, $A = \{1,2,3,4\}$ and $B = \{2,4,8,6,10\}$, and that the range may be from 1 to 10, the corresponding venn diagram is shown in Figure 12.1. The entire range of numbers, commonly referred to as the numbers of interest, is defined as the '**universal set**', $\mathbb{E}$. In this case, $\mathbb{E} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. These two sets, A and B, are represented by the two circles, with the members of the sets located inside the circles. It is clear that 2 and 4 are in both sets from this diagram, and that, 5, 7 and 9 don't belong to either set.

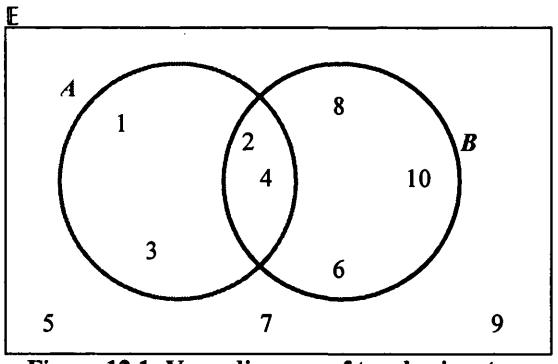


Figure 12.1: Venn diagram of two basic sets.

To describe if something is an element of a set or not, the following notation is used:

$1 \in A$	1 is an element of A
$2 \in A$	2 is an element of A
$6 \notin A$	6 is not an element of A
$5 \notin A$	5 is not an element of A

In some cases, sets are contained within another set. For example, given that the range of interest is whole numbers from 0 to 14 and the three sets:

$$A = \{x : x \text{ is positive, x is a whole number and } x \leq 12\} \quad (12.4)$$

$$B = \{x : x \text{ is positive, x is odd, x is a whole number and } x < 10\} \quad (12.5)$$

$$C = \{1,3,5,7,9,11\} \quad (12.6)$$

The venn diagram showing these sets is shown:

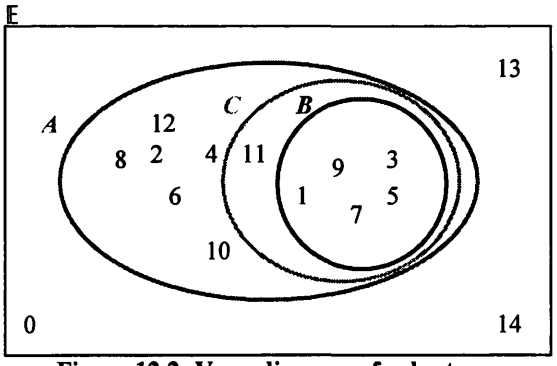


Figure 12.2: Venn diagram of subsets.

Clearly set C is contained within A and set B is contained in both C and A. Three symbols are used to describe this relationship:

$A \subseteq A$	A is a subset of A (i.e. A is contained in A).	(12.7)
$A \not\subset B$	A is not a subset of B (i.e. A is not contained within B).	(12.8)
$A \not\subset C$	A is not a subset of C (i.e. A is not contained in C).	(12.9)
$B \subset A$	B is a proper subset of A (i.e. all elements of B are contained within A).	(12.10)
$B \subseteq B$	B is a subset of B (i.e. B is contained in B).	(12.11)
$B \subset C$	B is a proper subset of C (i.e. all elements of B are contained within C).	(12.12)
$C \subset A$	C is a proper subset of A (i.e. all elements of C are contained within A).	(12.13)
$C \not\subset B$	C is not a subset of B (i.e. C is not contained in B).	(12.14)
$C \subseteq C$	C is a subset of C (C is contained in C).	(12.15)

A proper subset means that B is a proper subset of A if all elements of B are contained within A and that A contains elements that are not in B. Thus, a subset means that a set fits exactly inside another with all of the elements common to both sets. The dual of a subset is a superset, and hence the above could also be written as:

$A \supseteq A$	A is a superset of A (i.e. A is contained in A).	(12.16)
$A \supset B$	A is a proper superset of A (i.e. B is contained in A).	(12.17)
$A \supset C$	A is a proper superset of C (i.e. C is contained in A).	(12.18)
$A \supset B$	A is a proper superset of B (i.e. B is contained in A).	(12.19)
$B \supseteq B$	B is a superset of B (i.e. B is contained in B).	(12.20)
$C \supset B$	C is a superset of B (i.e. B is contained in C).	(12.21)
$A \supset C$	A is a proper superset of C (i.e. C is contained in A).	(12.22)
$C \supseteq C$	C is a superset of C.	(12.23)

12.1.2 Intersection

The intersection of two sets, A and B, is the elements contained within both sets. This is defined as given by equation 12.24. For example, given set $A = \{3,4,5,6\}$ and set $B = \{3,5,9,10,15\}$, the result of an intersection is given in equation 12.25.

$$A \cap B = \{x : x \in A \text{ and } x \in B\} \tag{12.24}$$

$$A \cap B = \{4,4,5,6\} \cap \{3,5,9,10,15\} = \{3,5\} \tag{12.25}$$

To make this clearer, a venn diagram is shown in Figure 12.3. The intersection is the area where the regions overlap. As can be seen, A and B overlap (have an intersection) as do B and C. However, A and B don't intersect and hence performing an intersection on these two sets would give an empty set. This means that the regions are not joint, commonly called '**disjoint**'. An intersection can be used to group common properties of sets together.

12.1.3 Union

The union of two sets, A and B, results in a set that contains the elements of both sets. This is defined by equation 12.26. This reads, the union is equal to x such that x is contained in set A or x is contained in set B. Note that x can be contained in both A and B. For example, given set $A = \{3,5,9,16\}$ and set $B = \{3,5,9,10,15\}$, the result of a union is given by equation 12.27.

$$A \cup B = \{x : x \in A \text{ or } x \in B\} \tag{12.26}$$

$$A \cup B = \{3,5,9,16\} \cup \{3,5,9,10,15\} = \{3,5,9,10,15,16\}. \tag{12.27}$$

Note that although 3, 5 and 9 are contained in both, they are only listed once in the result. From Figure 12.3, it is clear that the union of A and B would contain all elements inside of A and B.

12.1.4 Compliment

The compliment or inverse of a set can also be used to take advantage of the relationships between various functions. A compliment of set A consists of all elements not in A as shown in Figure 12.4. This is defined as:

$$\overline{A} = A^C = \{x : x \notin A\} \tag{12.28}$$

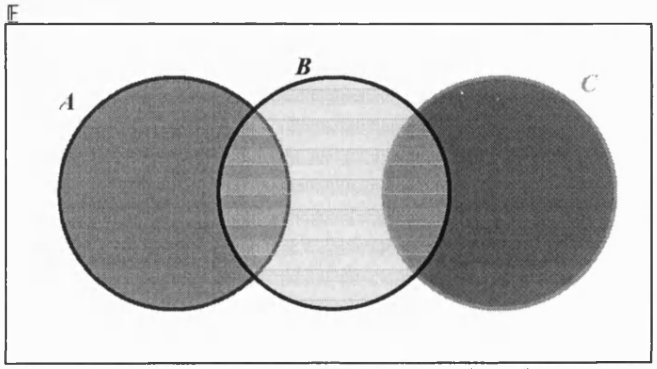


Figure 12.3: Venn diagram of the intersection of two sets.

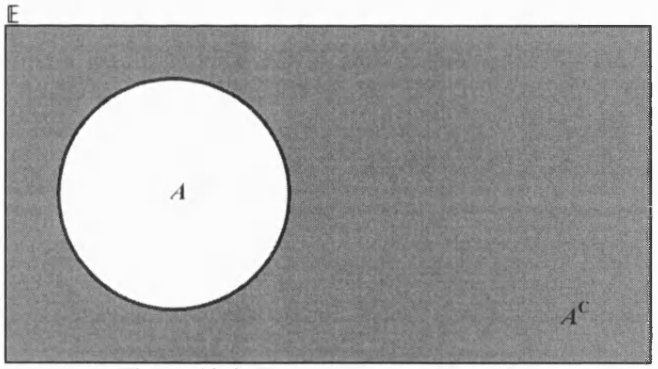


Figure 12.4: The compliment of a set.

12.1.5 Set Difference

The set difference can be useful when comparing different sets. A set difference between set A and B will contain all the elements that are in A but are not in B. This is defined as shown in equation 12.29. For example, given two sets, $A = \{3,2,5\}$ and $B = \{5,7\}$, the set difference is given by equation 12.30. Again this can be seen more clearly in a venn diagram as shown in Figure 12.5. The shaded area shows the set difference, A-B.

$$A - B = A / B = \{x : x \in A, x \notin B\} \tag{12.29}$$

$$A - B = \{3,2,5\} - \{5,7\} = \{2,3\} \tag{12.30}$$

There is another commonly used set difference equation called the '**Symmetric Set Difference**'. This results in a set containing the elements contained in either set but not both sets and is given by equation 12.31. Using the sets, $A = \{3,2,5\}$ and $B = \{5,7\}$, the set difference is given by equation 12.32. From Figure 12.5, it should be clear that the only bit of the two regions that is not in the symmetric set difference, is the overlapping (intersection) region.

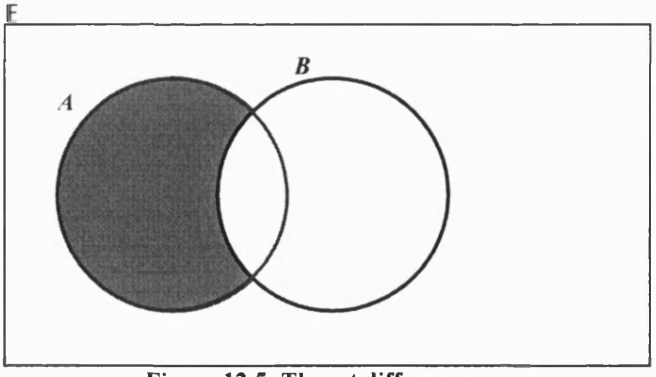


Figure 12.5: The set difference.

$$A\Delta B = \{x : x \in A, x \in B, x \notin (A \cap B)\} \tag{12.31}$$

$$A\Delta B = \{3,2,5\}\Delta\{5,7\} = \{2,3,7\} \tag{12.32}$$

12.1.6 Equality

Not all sets are equal. Set A is equal to another set, B, if they both contain exactly the same elements. For example, given two sets, $A = \{14,2,10\}$ and $B = \{10,14,2\}$, since they both contain the same elements, regardless of order, then A is equal to B (i.e. A=B). The number of times an element is repeated is also irrelevant. If set B was given as, $B = \{10,14,14,2,2,2\}$, A is still equal to B as it still contains the same elements. However, if B was set to, $B = \{2,14,10,10,3\}$, then this is not equal to A as it contains elements not in A (i.e. $A \neq B$).

12.1.7 Translation

A set such as an image can be moved by the use of translation as given by equation 12.33. For example, given an a set and that $i = (x, y)$, the translation simply moves the set by x in the x-axis and by y in the y-axis as shown in Figure 12.6.

$$A_i = (A)_i = \{b : b = a + i, \text{for a} \in \text{A}\} \tag{12.33}$$

12.1.8 Reflection

The reflection of a set is a useful operator, especially for morphology and is given by equation 12.34. This simply flips a set about the origin in the x and y axis as shown in Figure 12.7.

$$\hat{A} = \{x : x = -a, \text{for a} \in \text{A}\} \tag{12.34}$$

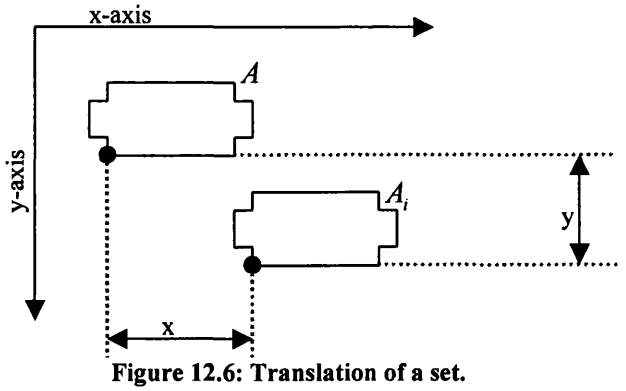


Figure 12.6: Translation of a set.

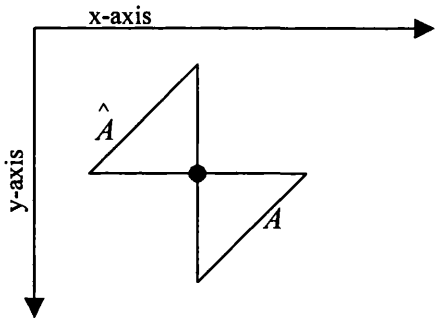


Figure 12.7: Reflection of a set.

12.2 Predefined Sets

There are some common sets of numbers used through out set theory. The definitions for these are:

$$\mathbb{Z} = \{\text{all whole numbers}\} = \{\cdots,-3,-2,-1,0,1,2,3,\cdots\} \tag{12.35}$$

$$\mathbb{N} = \{\text{all non - negative whole numbers}\} = \{x : x \geq 0, x \in Z\} = \{0,1,2,3,4,5,6,\cdots\} \tag{12.36}$$

$$\mathbb{N}^+ = \{\text{positive whole numbers}\} = \{x : x > 0, x \in Z\} = \{1,2,3,4,5,6,\cdots\} \tag{12.37}$$

$$\mathbb{R} = \{\text{all real numbers}\} = \{\cdots,-0.2,-0.1,0,0.1,0.2,\cdots\} \tag{12.38}$$

$$\mathbb{R}^+ = \{\text{all positive real numbers}\} = \{x : x > 0, x \in R\} = \{\cdots,0.1,0.2,0.3,0.4,\cdots\} \tag{12.39}$$

$$\mathbb{R}^- = \{\text{all negative real numbers}\} = \{x : x < 0, x \in R\} = \{\cdots,-0.3,-0.2,-0.1,\cdots\} \tag{12.40}$$

$$\mathbb{Q}^- = \{\text{all rational numbers}\} = \left\{x : x = \frac{p}{q}, p \in Z, q \in Z, q \neq 0\right\} = \left\{\cdots,-\frac{1}{2},-\frac{1}{4},\frac{1}{2},\frac{1}{4},\cdots\right\} \tag{12.41}$$

These are used in the same way as previously shown. For example:

$$A = \{x : x \in \mathbb{R} \text{ and } x < 2\} \tag{12.42}$$

This reads as; A is a set of values of x such that x is a real number (R) and x is less than 2. It should be clear from the above, that the following relationships are apparent; $\mathbb{N} \subseteq \mathbb{Z}$, $\mathbb{N} \subseteq \mathbb{R}$, $\mathbb{Z} \subseteq \mathbb{R}$, $\mathbb{Q} \cup \mathbb{T} = \mathbb{R}$, and $\mathbb{Q} \cap \mathbb{T} = \phi$.

12.3 Basic Laws

There are some basic laws that are used in set theory, which allow easy manipulation of set theory equations:

$$\left.\begin{aligned} A \cup B &= B \cup A \\ A \cap B &= B \cap A \end{aligned}\right\} \text{Commutative Laws} \tag{12.43}$$

$$\left.\begin{aligned} A\cup(B\cup C)&=(A\cup B)\cup C\\ A\cap(B\cap C)&=(A\cap B)\cap C\end{aligned}\right\}\text{Associative Laws}\qquad(12.44)$$

$$\left.\begin{aligned} A\cap(B\cup C)&=(A\cap B)\cup(A\cap C)\\ A\cup(B\cap C)&=(A\cup B)\cap(A\cup C)\end{aligned}\right\}\text{Distributive Laws}\qquad(12.45)$$

$$\left.\begin{aligned} A\cup\varnothing&=A\\ A\cap \mathrm{E}&=A\end{aligned}\right\}\text{Identity Laws}\qquad(12.46)$$

$$\left.\begin{aligned} A\cup\overline{A}&=\mathrm{E}\\ A\cap\overline{A}&=\mathrm{E}\\ \overline{\overline{A}}&=A\end{aligned}\right\}\text{Compliment Laws}\qquad(12.47)$$

Another set of laws can be derived from the above, which can be used to simplify and reduce set theory equations:

$$\left.\begin{aligned} A\cup(A\cap B)&=A\\ A\cap(A\cup B)&=A\end{aligned}\right\}\text{Absorption Laws}\qquad(12.48)$$

$$\left.\begin{aligned} (A\cap B)\cup(A\cap\overline{B})&=A\\ (A\cup B)\cap(A\cup\overline{B})&=A\end{aligned}\right\}\text{Minimisation Laws}\qquad(12.49)$$

$$\left.\begin{aligned} \overline{A\cup B}&=\overline{A}\cap\overline{B}\\ \overline{A\cap B}&=\overline{A}\cup\overline{B}\end{aligned}\right\}\text{De Morgan's Laws}\qquad(12.50)$$

12.4 Other useful operators

There are numerous set operators, some of the more useful of which are briefly described below.

$\exists x$	There exists x such that...
$\forall x$	For all x.
$\Rightarrow$	Implies.
$\Leftrightarrow$, *iff*	If and only if
Q.E.D	Quod erat demonstrandum (which was to be proved)
λX	Set X with a scaling factor λ. The scaled set is defined as: $\lambda X=\left\{x:\frac{x}{\lambda}\in X\right\}$
$\Psi(X)$	Transformation of set X with transform Ψ.
$\Psi^*(X)$	Dual transformation of set X (i.e. $\Psi^*(X)=\left[\Psi\left(X^C\right)\right]^C$).
$\bigcup_{b\in B} X_b$	Union of all of the translations X_b.
$\bigcap_{b\in B} X_b$	Intersection of all of the translations X_b.

13 Appendix B – Full Results

This appendix contains the complete set of results carried out for chapters 7 and 9.

13.1 Image Noise Reduction

This section contains the full results for the Barbara 2, Boats and Goldhill images from section 7.1.

13.1.1 Gaussian filtering results

This section contains the results for the Barbara 2, Boats and Goldhill images for Gaussian filtering of noisy images (see section 7.1.1). The results show how the PSNR varies as sigma and the mask size are varied (see Figures 13.1 – 13.12) and Table 13.1 shows the optimum filters and their corresponding sigma and mask sizes.

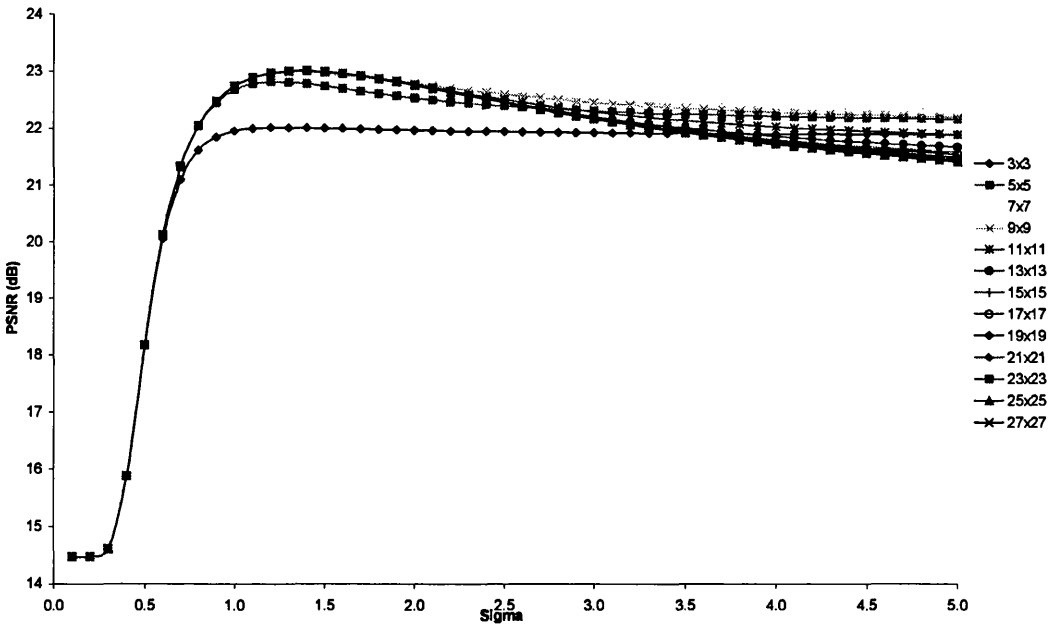


Figure 13.1: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale.

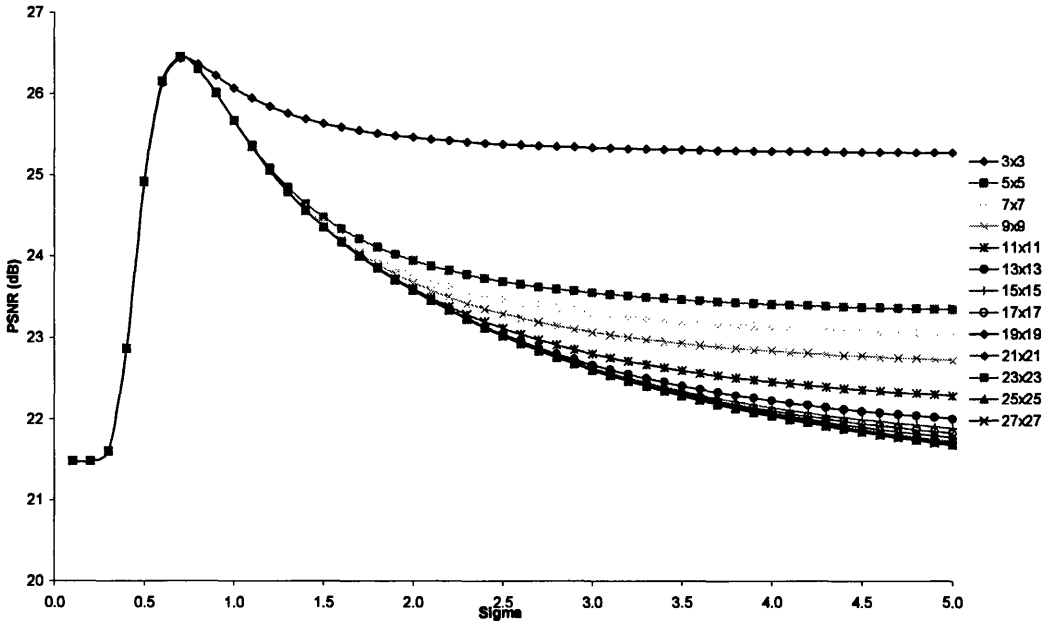


Figure 13.2: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale.

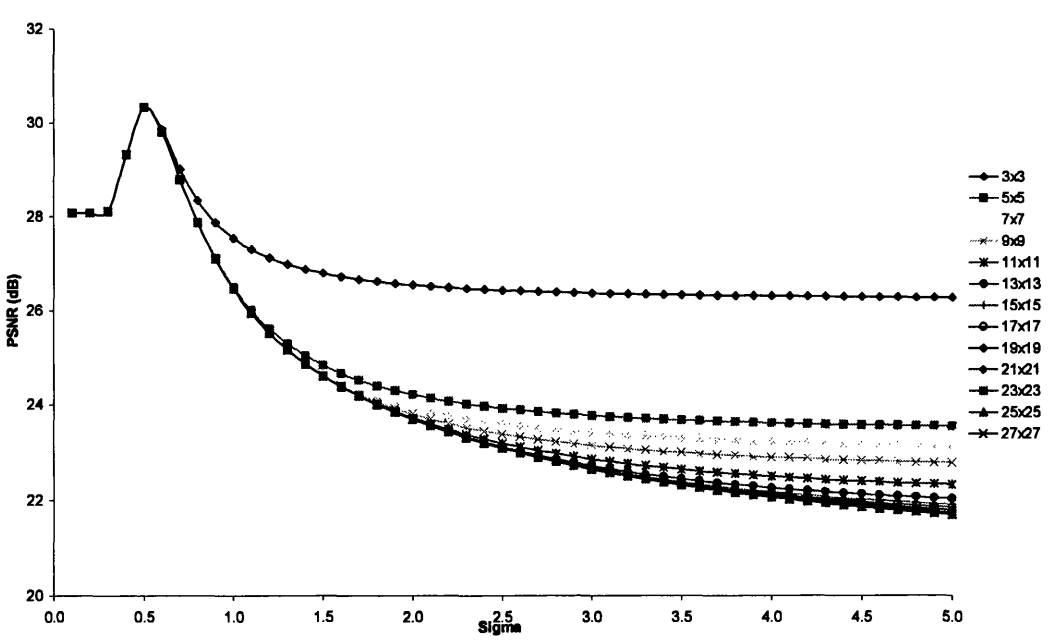


Figure 13.3: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale.

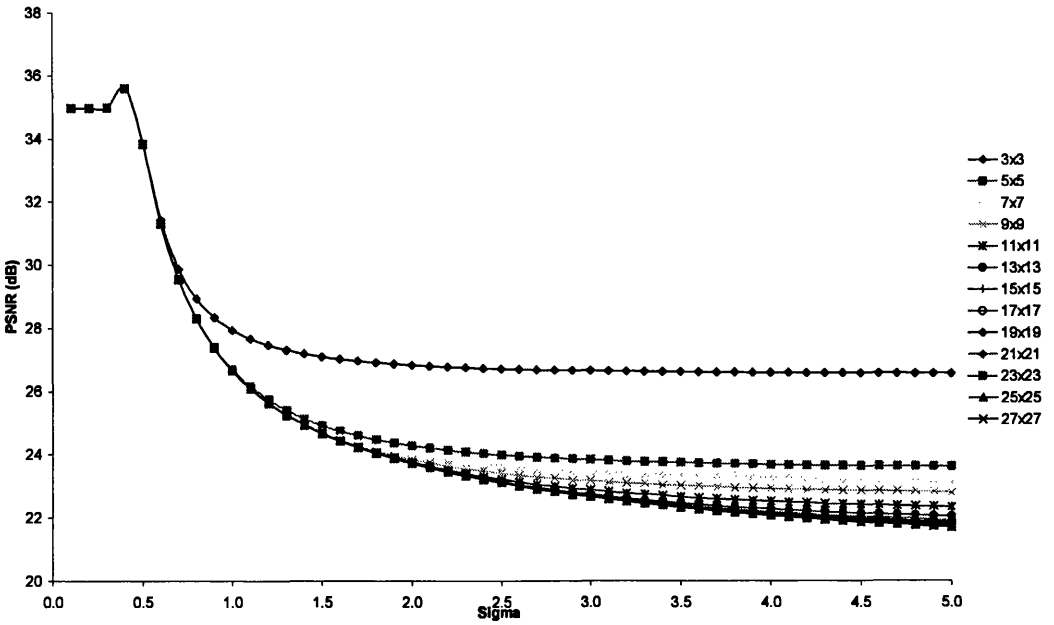


Figure 13.4: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Barbara 2 image, which has a size of 720 x 576 pixels and is 8bit greyscale.

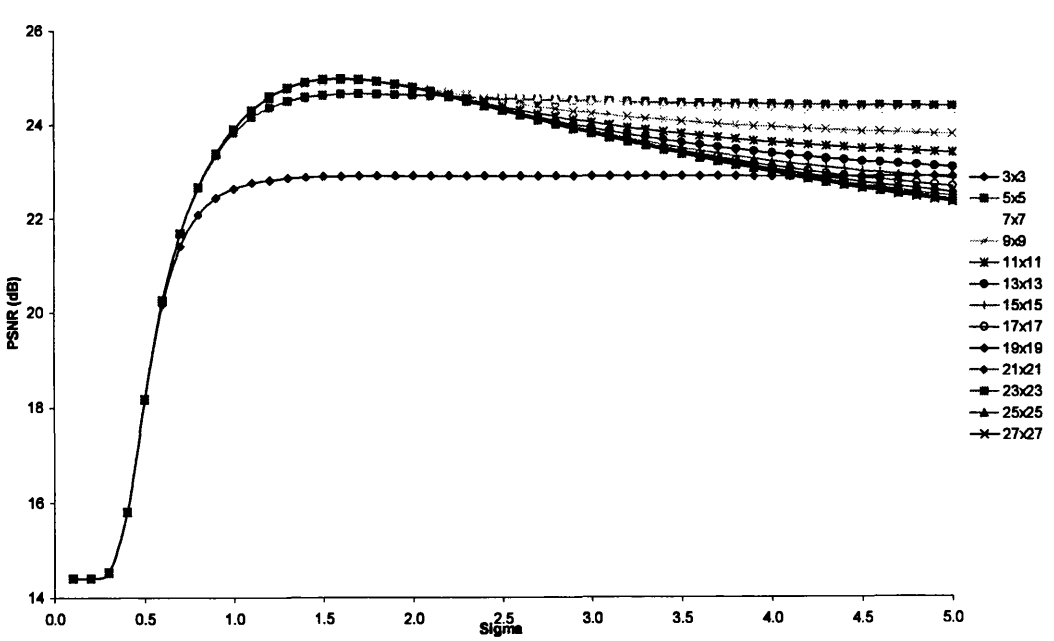


Figure 13.5: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale.

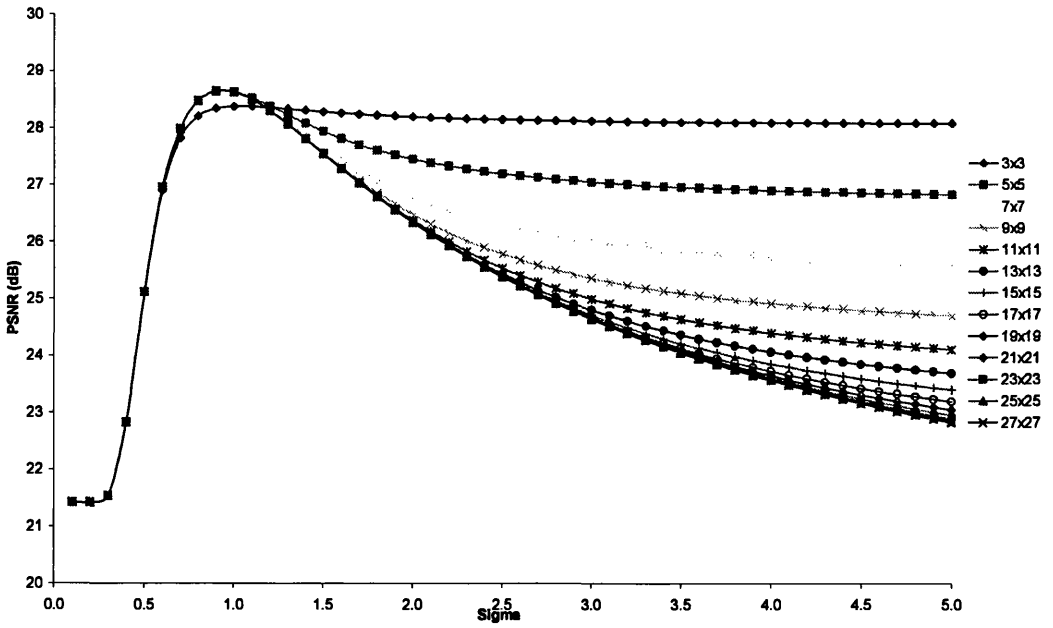


Figure 13.6: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale.

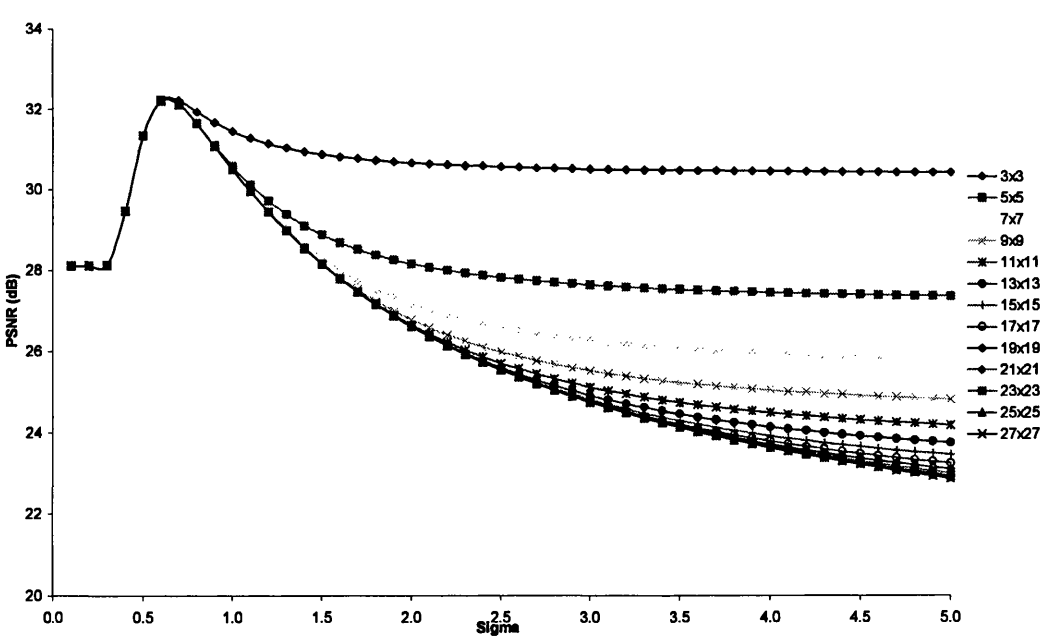


Figure 13.7: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale.

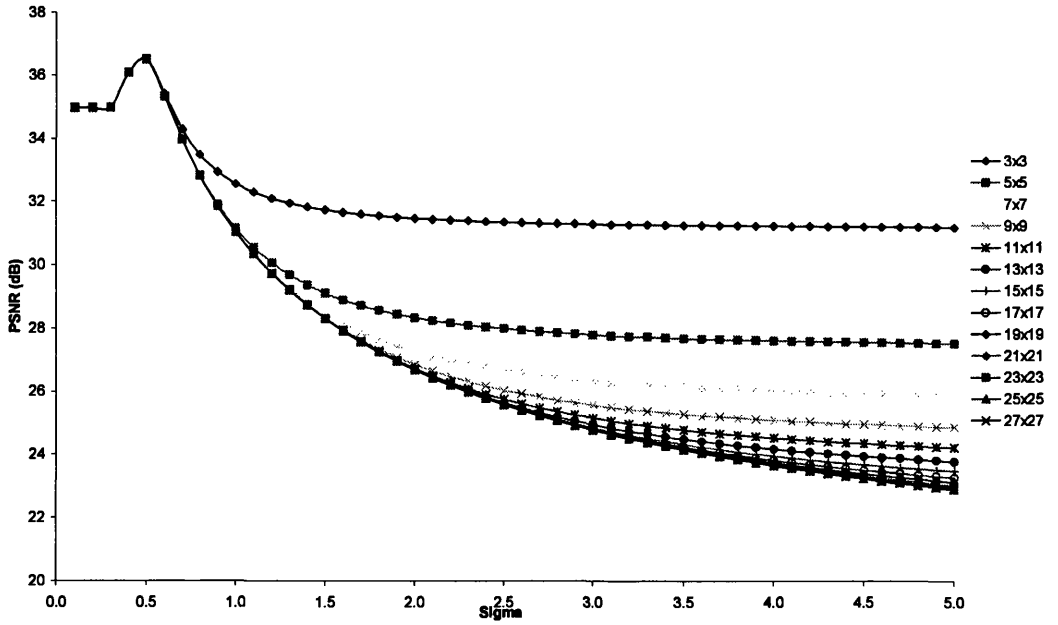


Figure 13.8: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Boats image, which has a size of 720 x 576 pixels and is 8bit greyscale.

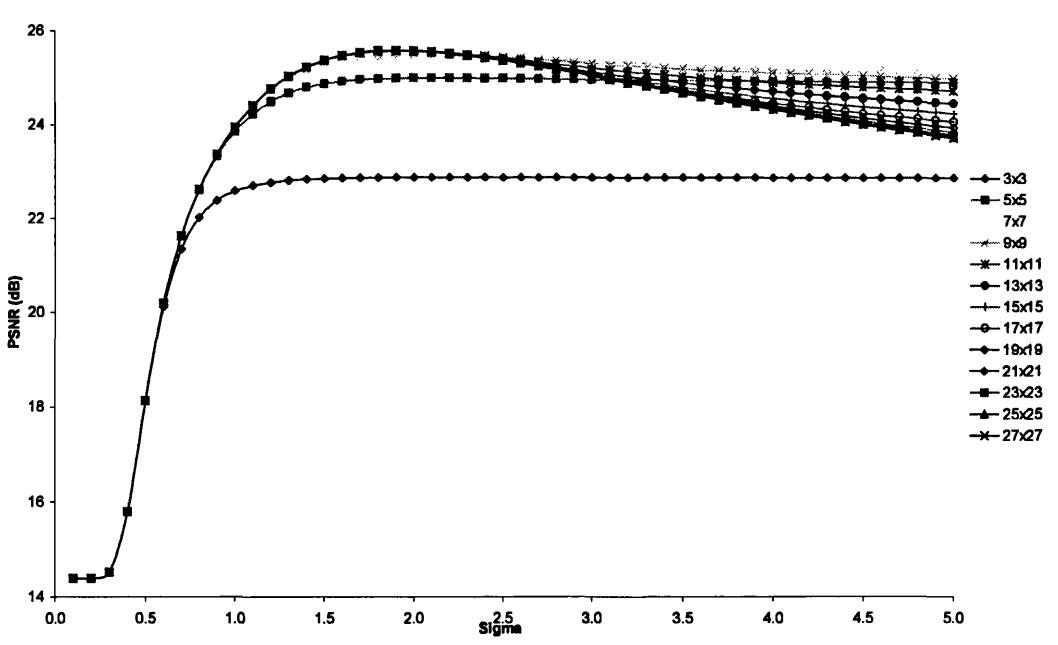


Figure 13.9: Gaussian results showing PSNR against sigma and mask size with 14dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale.

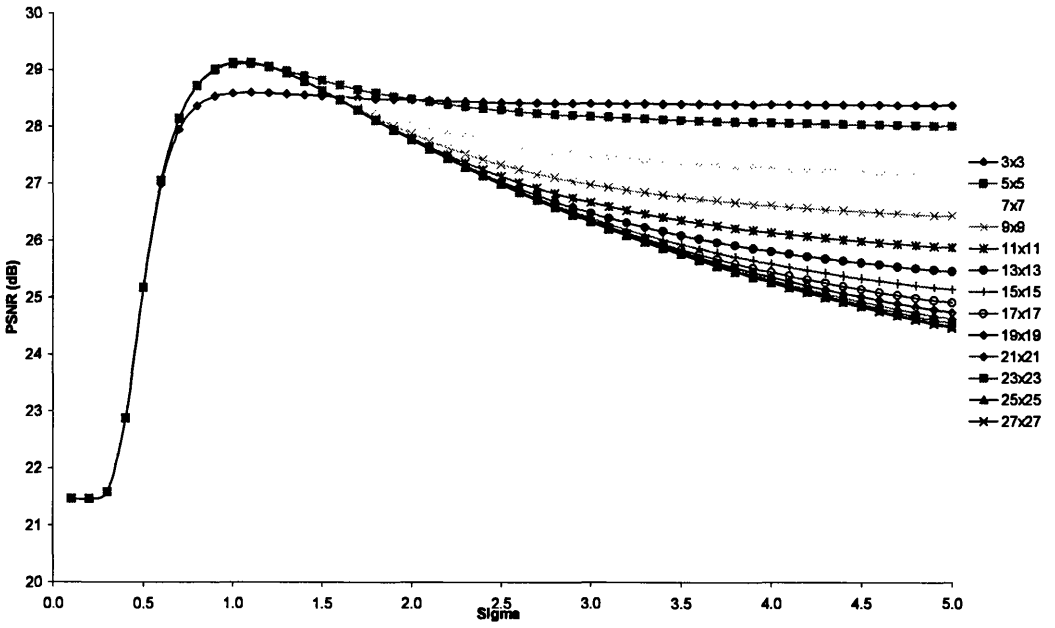


Figure 13.10: Gaussian results showing PSNR against sigma and mask size with 21dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale.

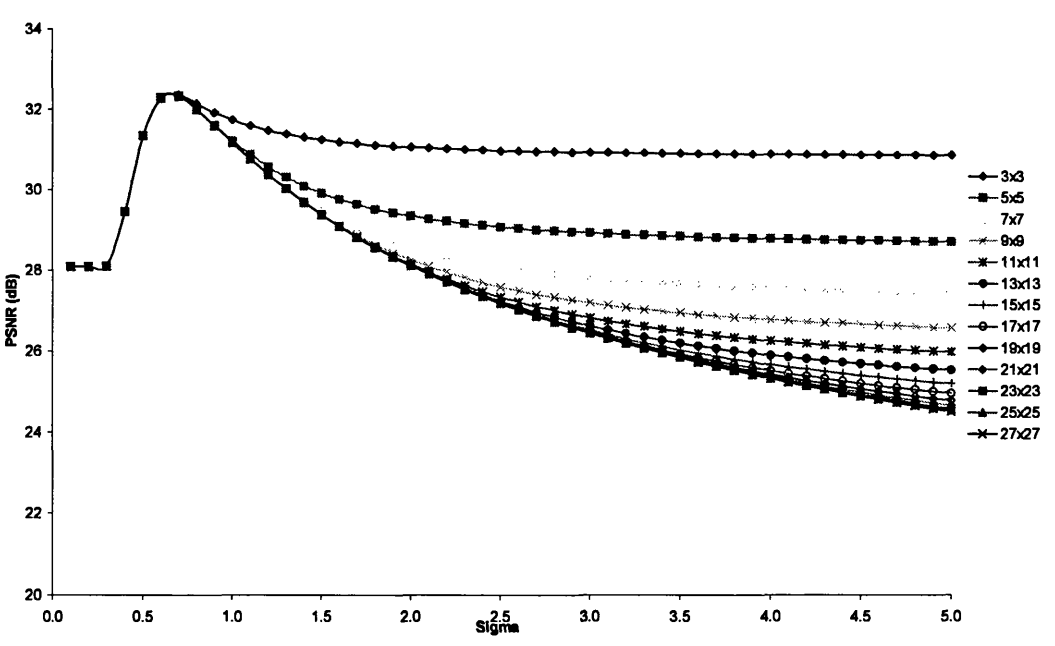


Figure 13.11: Gaussian results showing PSNR against sigma and mask size with 28dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale.

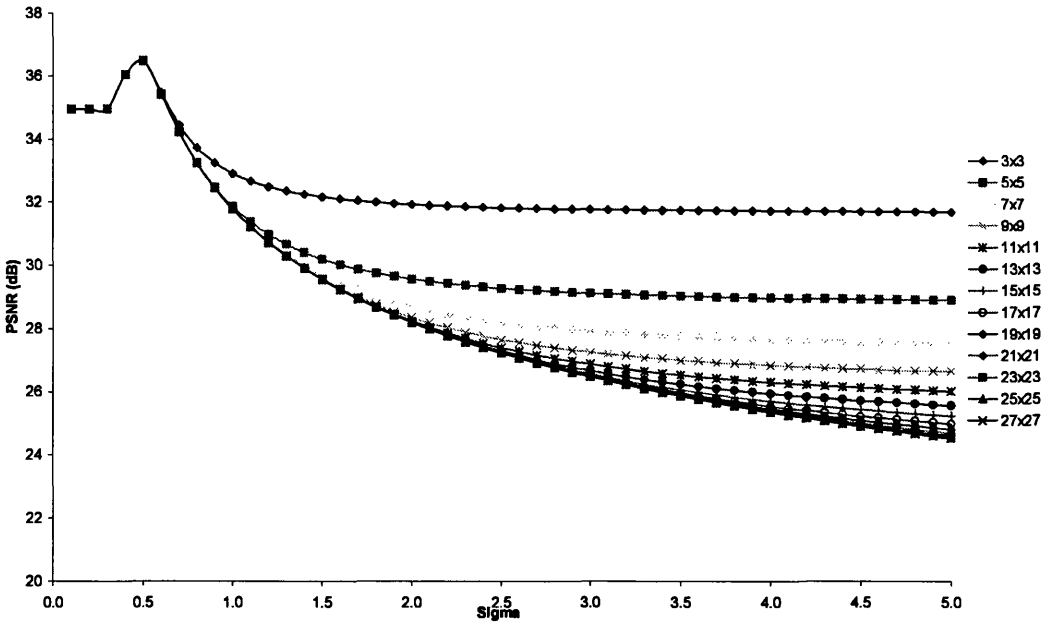


Figure 13.12: Gaussian results showing PSNR against sigma and mask size with 35dB of noise using the Goldhill image, which has a size of 720 x 576 pixels and is 8bit greyscale.

	Sigma	Mask Size	PSNR (dB)
Barbara 2 14.47dB	1.4	15 x 15	23.01
Boats 14.41dB	1.6	13 x 13	24.99
Goldhill 14.39dB	1.9	15 x 15	15.58
Barbara 2 21.49dB	0.7	7 x 7	26.46
Boats 21.43dB	0.9	9 x 9	28.66
Goldhill 21.47dB	1.1	9 x 9	29.14
Barbara 2 28.08dB	0.5	3 x 3	30.34
Boats 28.11dB	0.6	5 x 5	32.23
Goldhill 28.09dB	0.7	3 x 3	32.35
Barbara 2 34.97dB	0.4	3 x 3	35.60
Boats 34.99dB	0.5	3 x 3	36.52
Goldhill 34.96dB	0.5	3 x 3	36.49

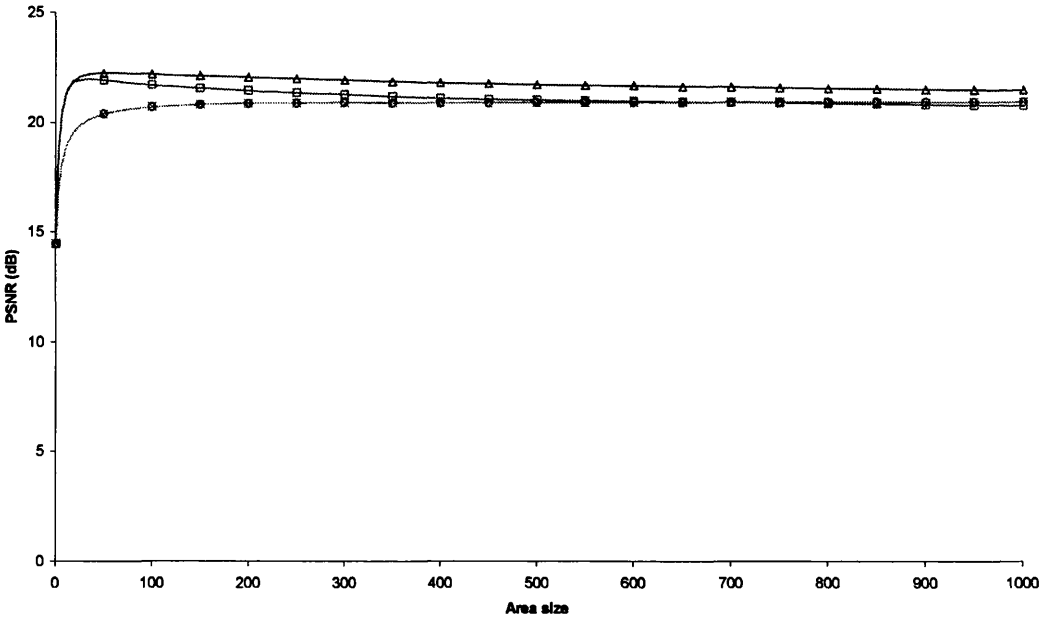
Table 13.1: The optimum sigma values and mask sizes for noise removal using the Gaussian filter and their performance gain in PSNR.

Image and noise level (dB)	PSNR (dB)			
	Un-filtered images		Optimal filtered images	
	JPEG	JPEG 2000	JPEG	JPEG 2000
Barbara 2 (Original noise free)	27.66	30.64		
Boats (Original noise free)	32.66	35.00		
Goldhill (Original noise free)	31.39	32.83		
Barbara 2 14.47dB	18.64	17.48	21.98	22.18
Boats 14.41dB	18.58	17.87	21.91	21.92
Goldhill 14.39dB	15.51	17.73	25.29	25.52
Barbara 2 21.49dB	23.95	24.42	24.86	25.60
Boats 21.43dB	25.41	24.33	21.82	21.85
Goldhill 21.47dB	25.11	24.22	27.83	28.32
Barbara 2 28.08dB	26.78	28.65	26.76	28.23
Boats 28.11	29.98	30.89	23.05	23.13
Goldhill 28.09dB	29.42	29.92	29.92	30.59
Barbara 2 34.97dB	27.53	30.13	27.16	29.14
Boats 34.99dB	32.06	34.17	21.49	21.44
Goldhill 34.96dB	30.89	32.36	29.58	30.61

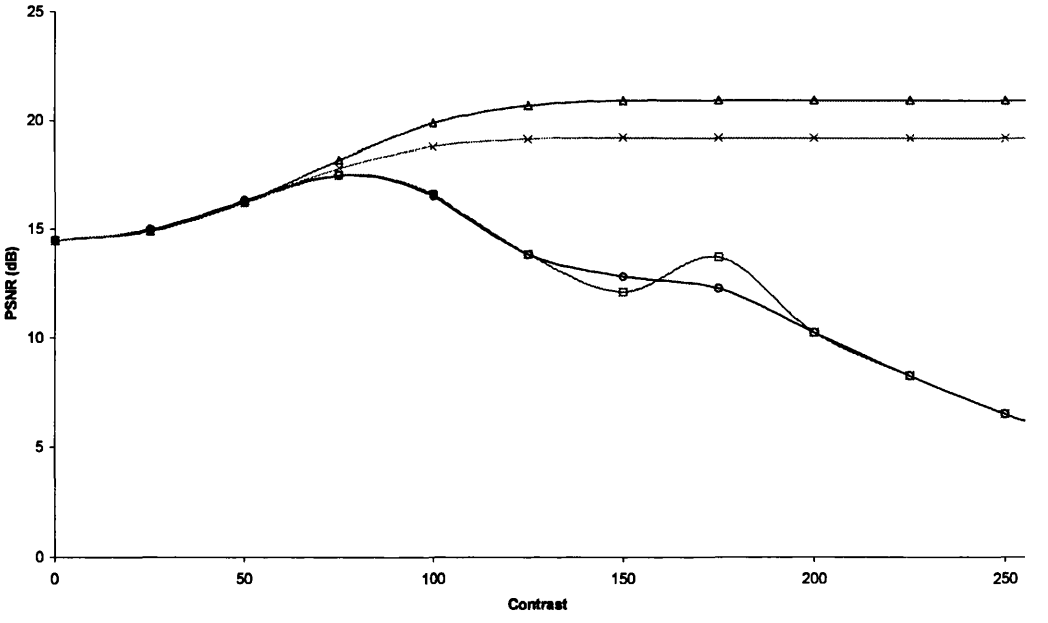
Table 13.2: PSNR values for the output of the JPEG and JPEG 2000 CODECs using the optimum Gaussian filtered images as the input. The CODEC outputs are compared to the original noise free image.

13.1.2 Morphology filtering results

This section contains the additional results obtained in section 7.1.2.



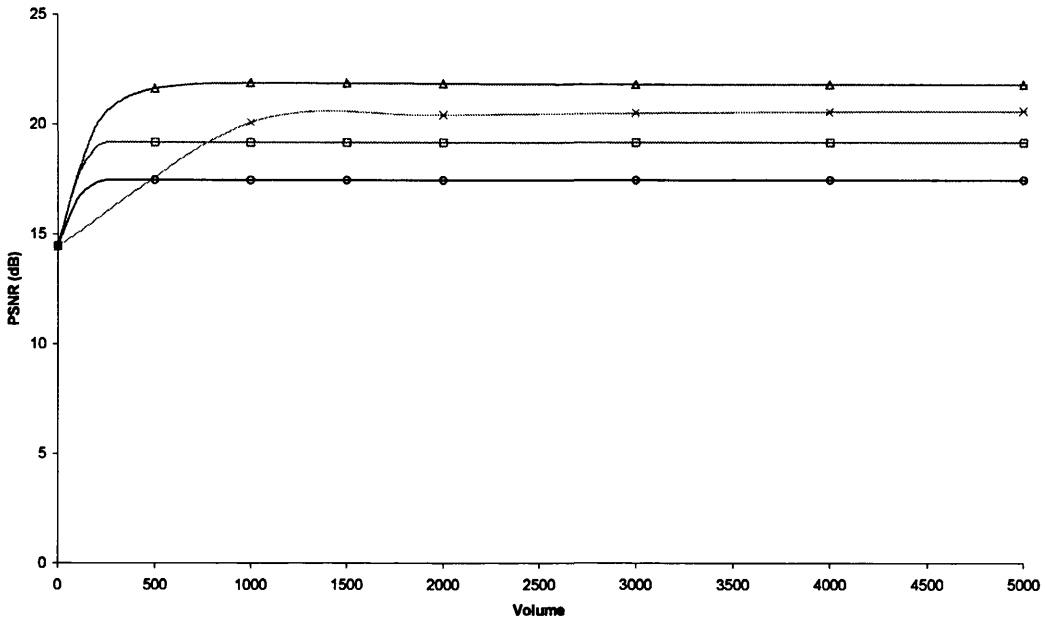
a) Results for the area attribute.



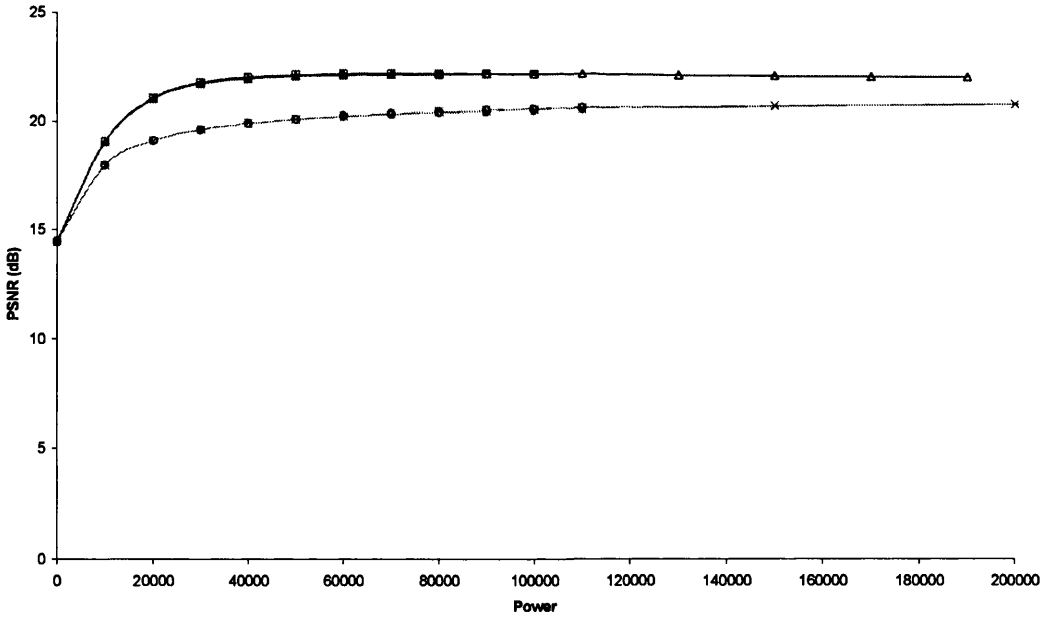
b) Results for the contrast attribute.

Key: 4nmAF 4nmASF 8nmAF 8nmASF

Figure 13.13: Area and contrast attribute results for the Barbara 2 image with 14dB of noise.



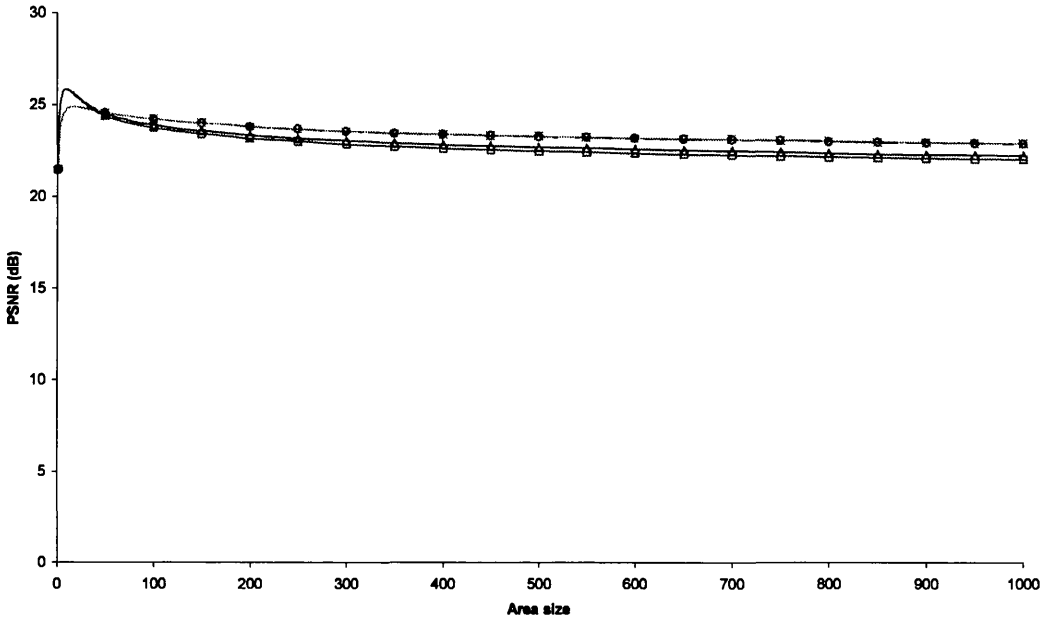
a) Results for the volume attribute.



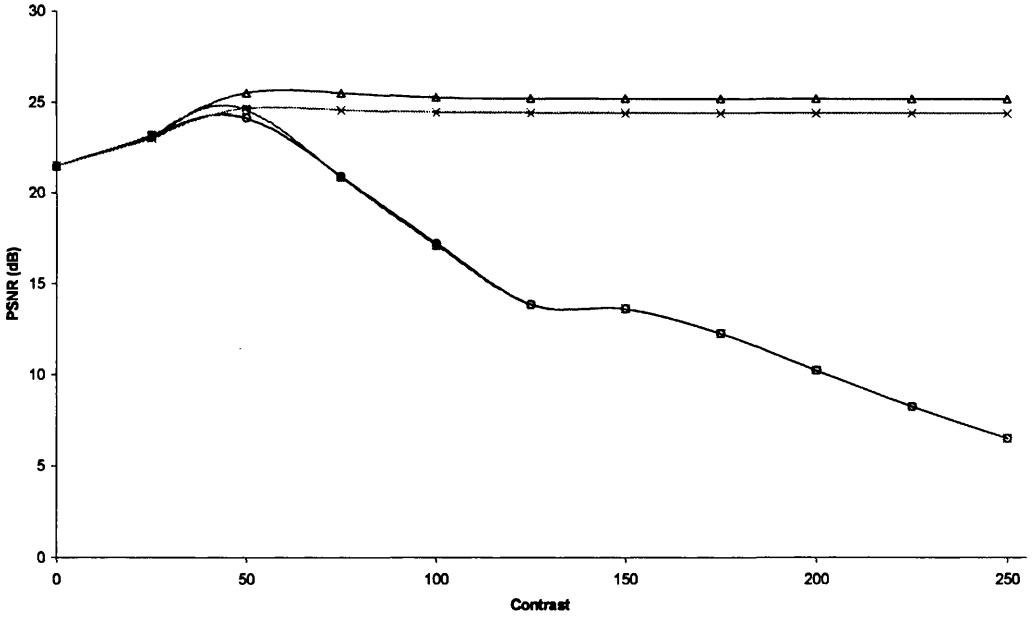
b) Results for the power attribute.

Key: 4nnAF 4nnASF 8nnAF 8nnASF

Figure 13.14: Volume and power attribute results for the Barbara 2 image with 14dB of noise.



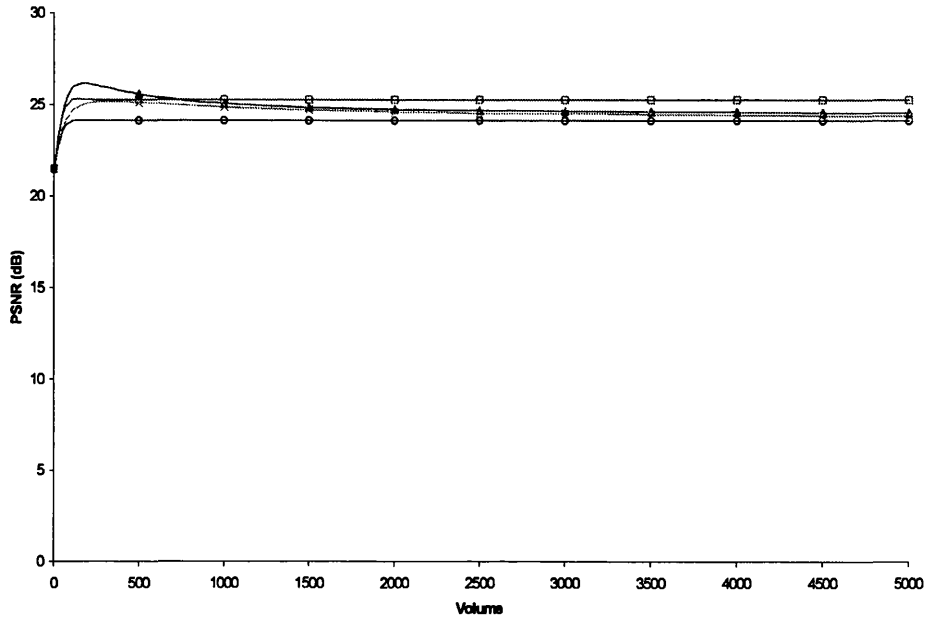
a) Results for the area attribute.



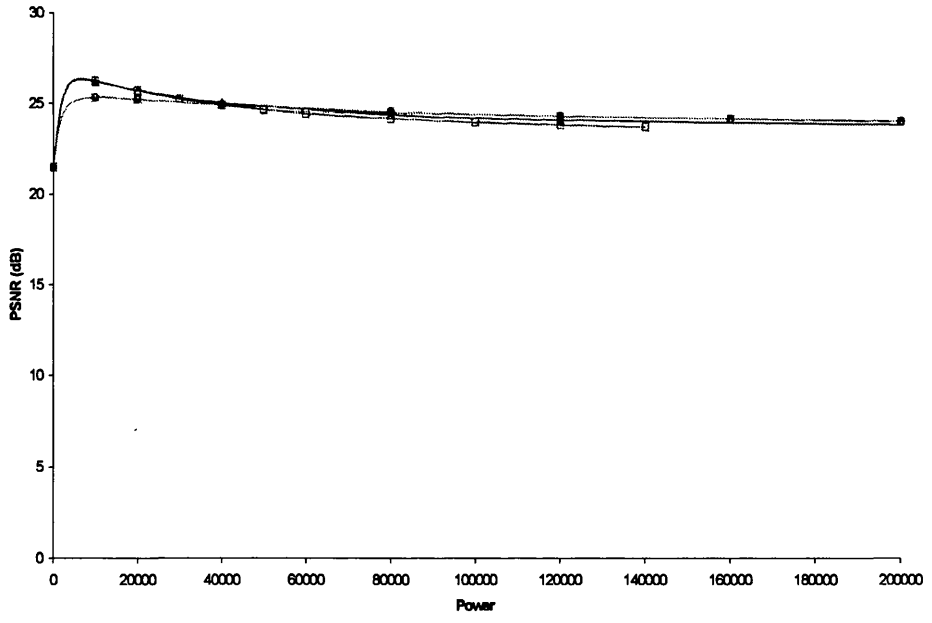
b) Results for the contrast attribute.

Key: 4nnAF 4nnASF 8nnAF 8nnASF

Figure 13.15: Area and contrast attribute results for the Barbara 2 image with 21dB of noise.



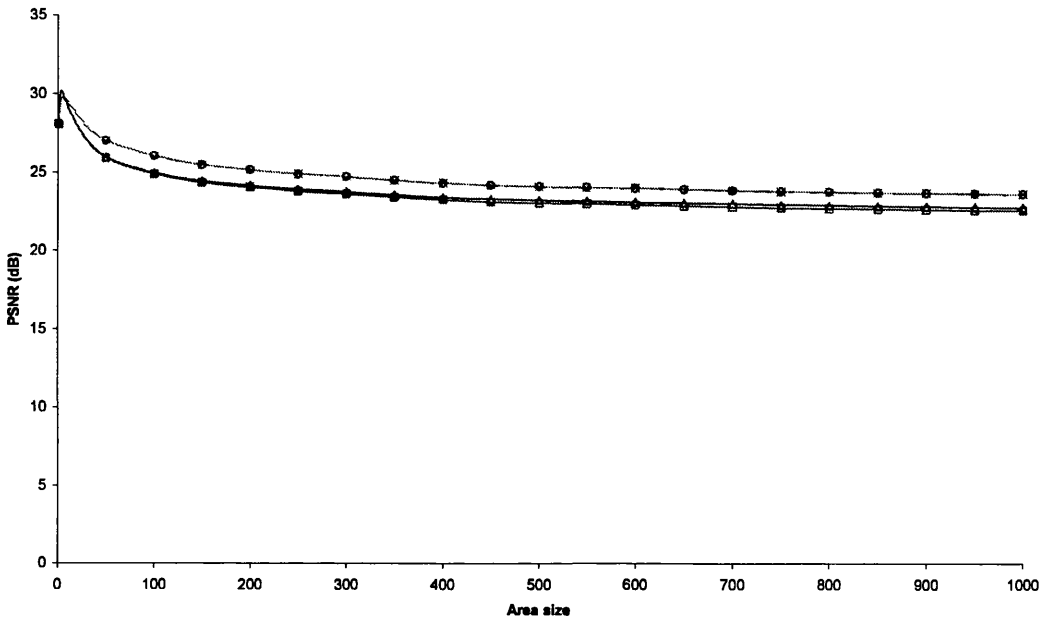
a) Results for the volume attribute.



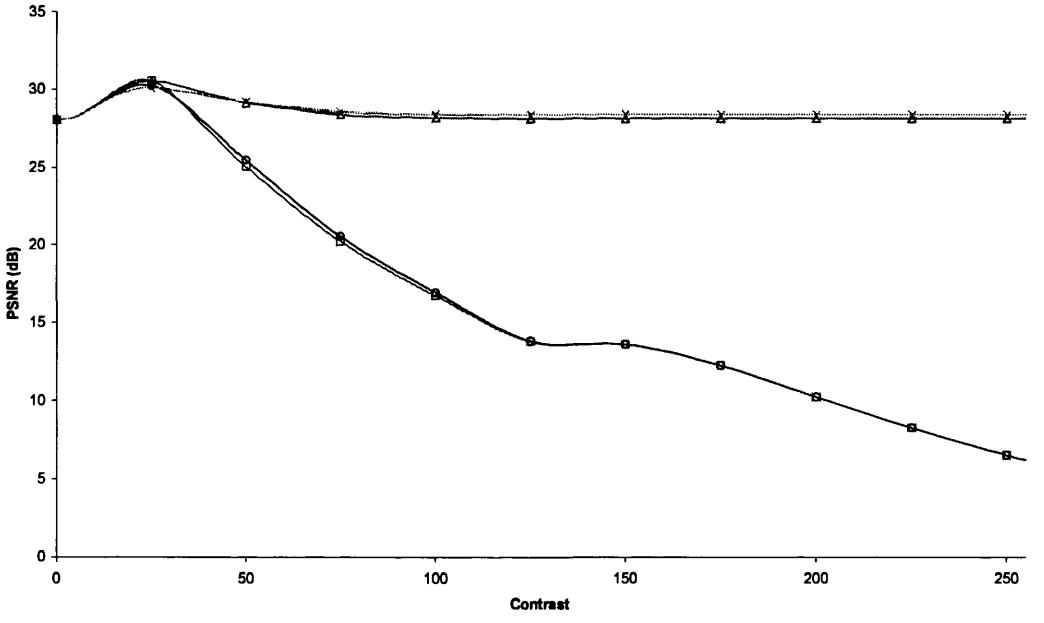
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.16: Volume and power attribute results for the Barbara 2 image with 21dB of noise.



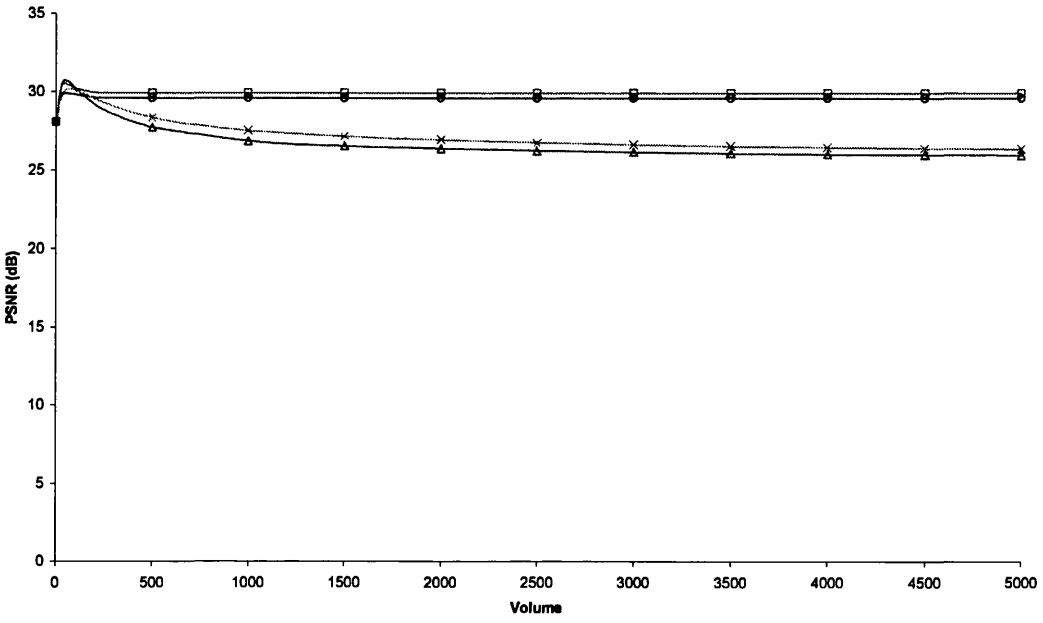
a) Results for the area attribute.



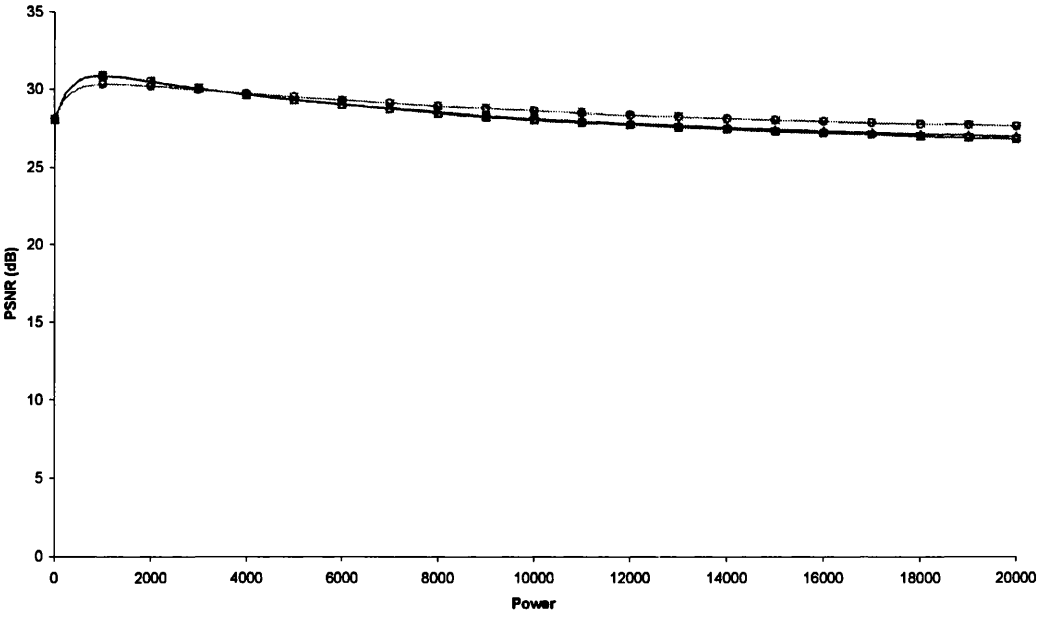
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.17: Area and contrast attribute results for the Barbara 2 image with 28dB of noise.



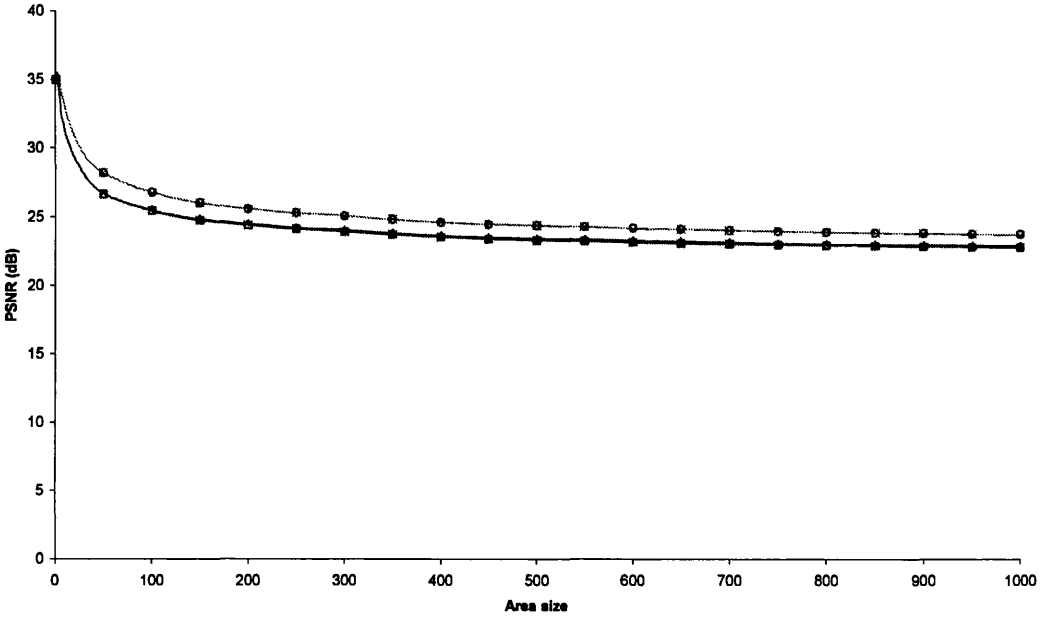
a) Results for the volume attribute.



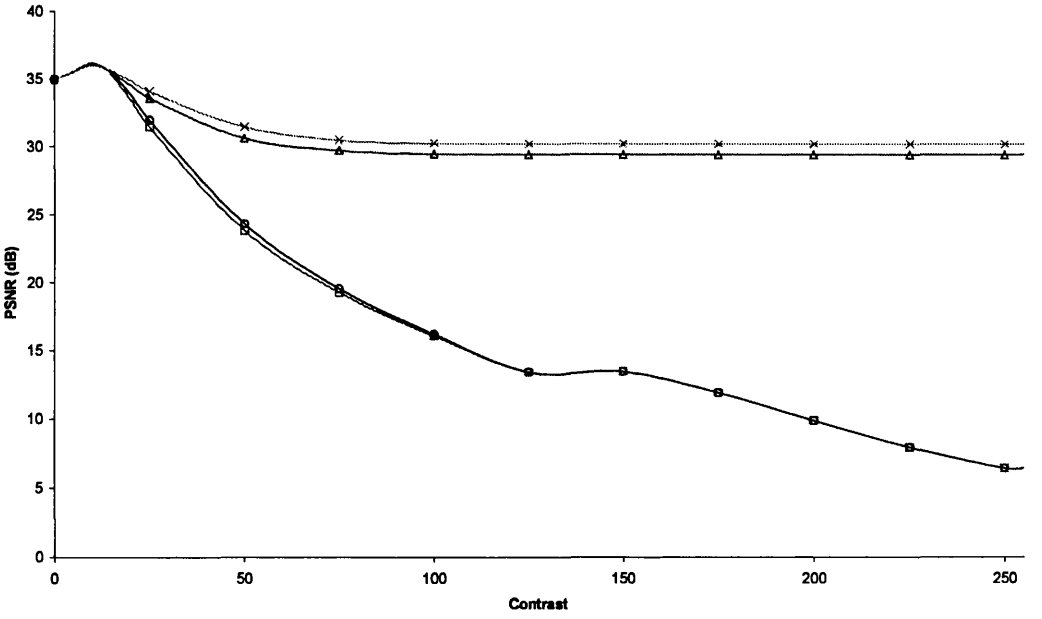
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.18: Volume and power attribute results for the Barbara 2 image with 28dB of noise.



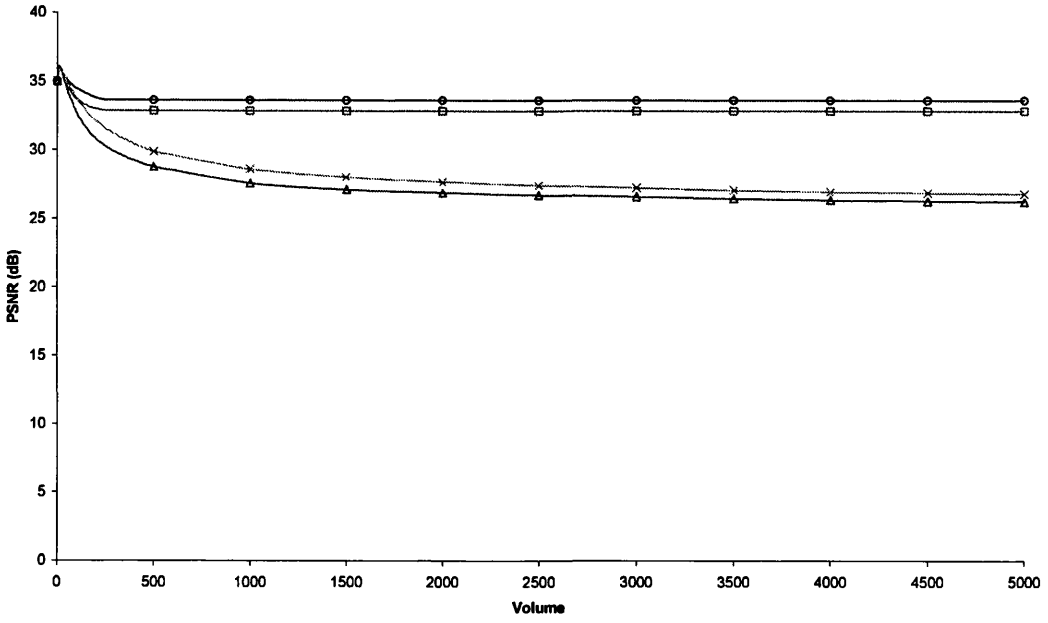
a) Results for the area attribute.



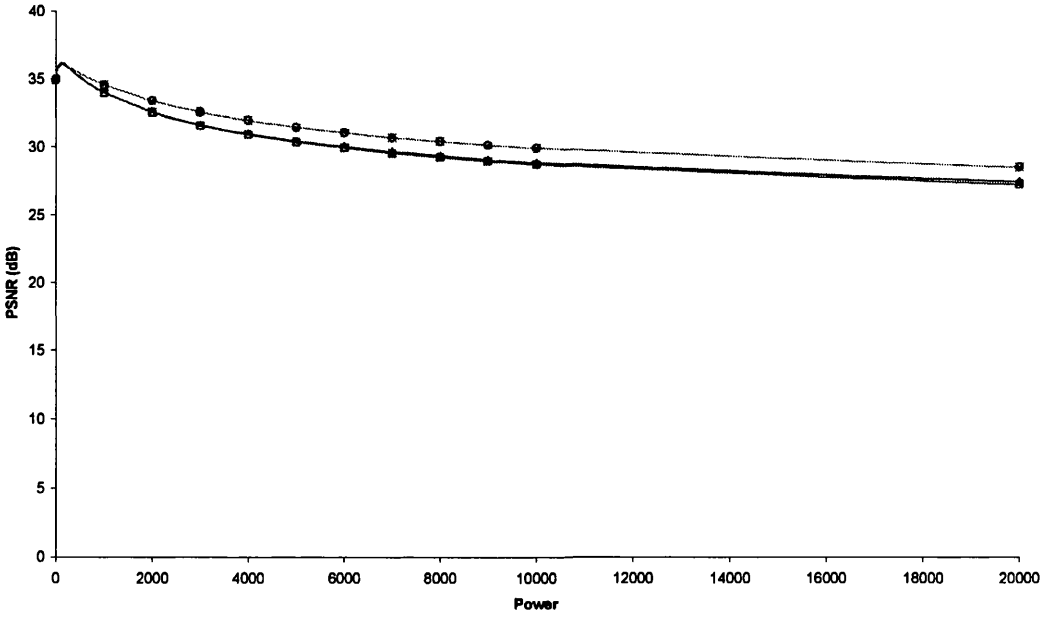
b) Results for the contrast attribute.

Key: 4nmAF 4nmASF 8nmAF 8nmASF

Figure 13.19: Area and contrast attribute results for the Barbara 2 image with 35dB of noise.



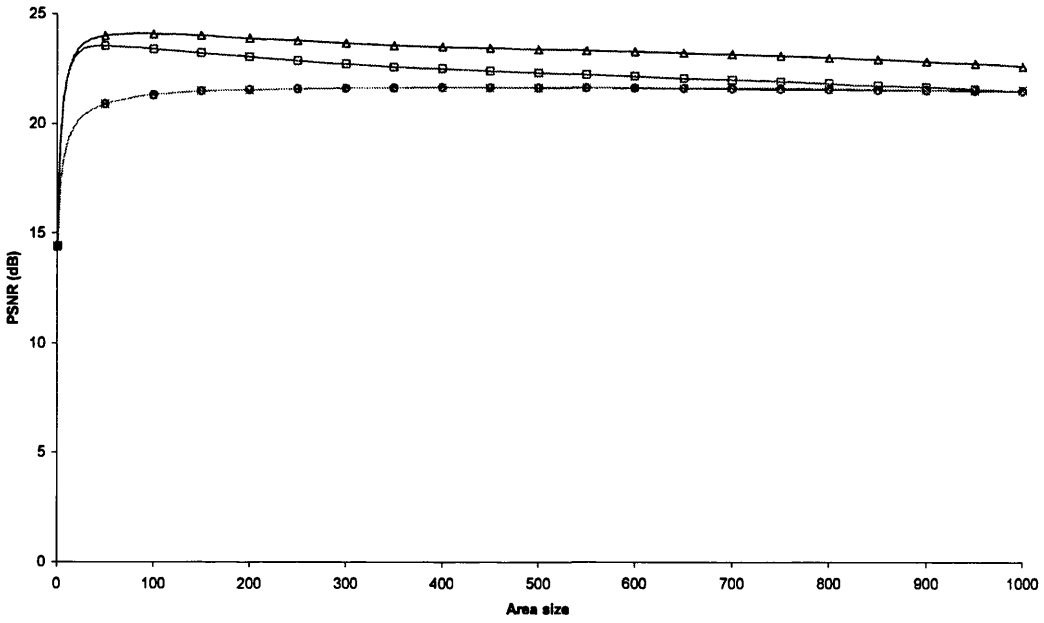
a) Results for the volume attribute.



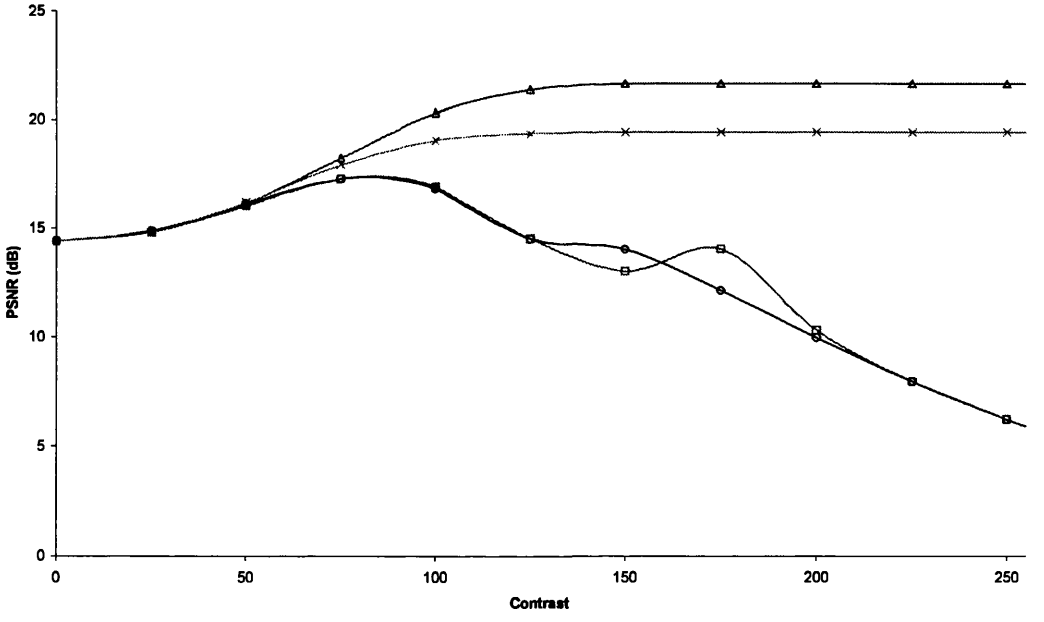
b) Results for the contrast attribute.

Key: 4nnAF 4nnASF 8nnAF 8nnASF

Figure 13.20: Volume and power attribute results for the Barbara 2 image with 35dB of noise.



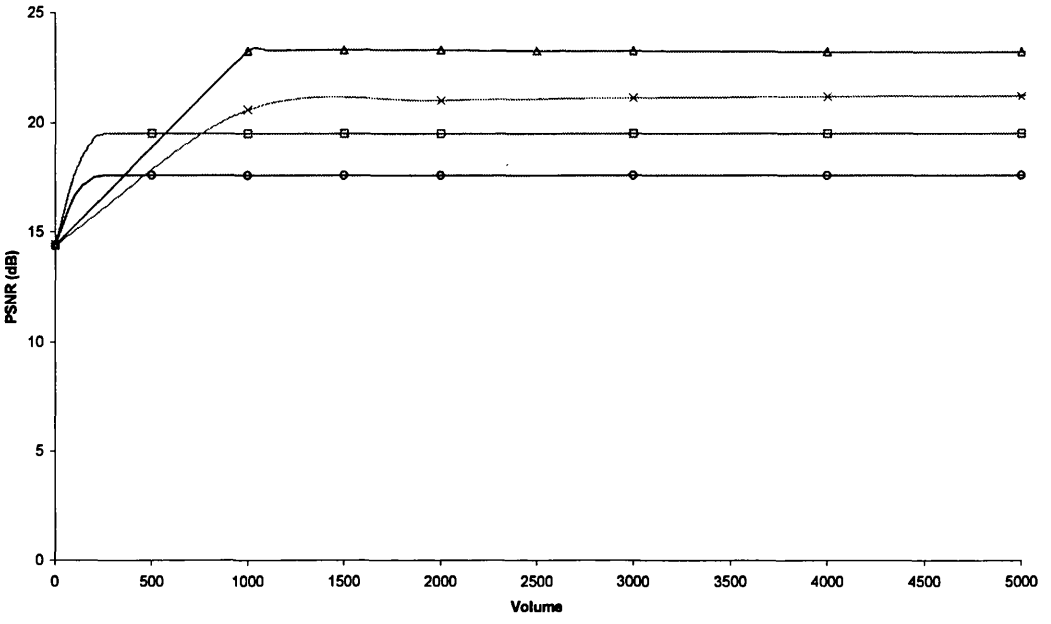
a) Results for the area attribute.



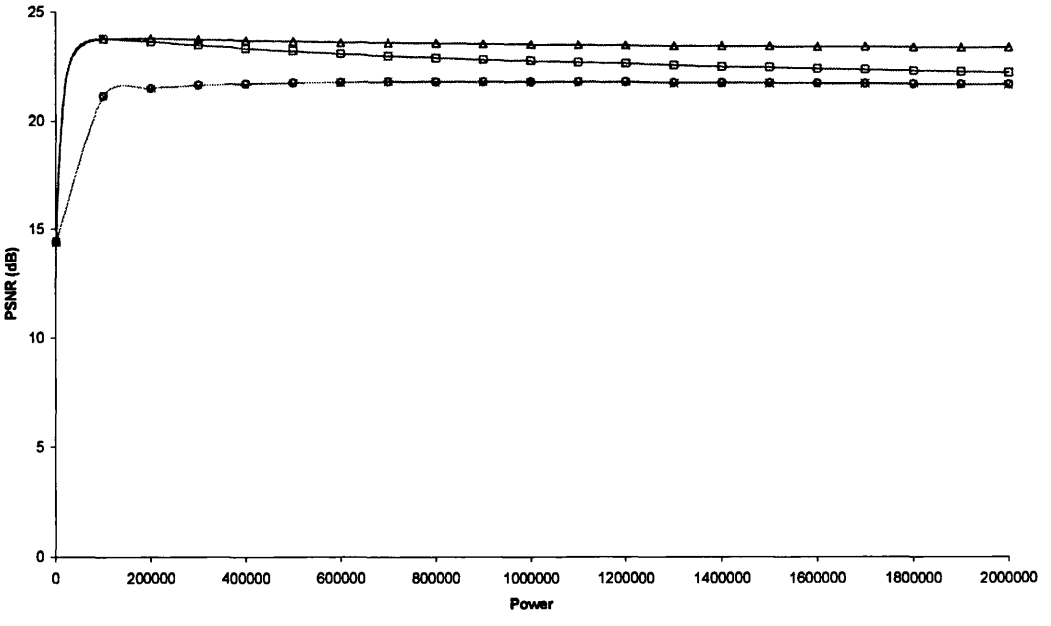
b) Results for the contrast attribute.

Key: 4nnAF 4nnASF 8nnAF 8nnASF

Figure 13.21: Area and contrast attribute results for the Boats image with 14dB of noise.



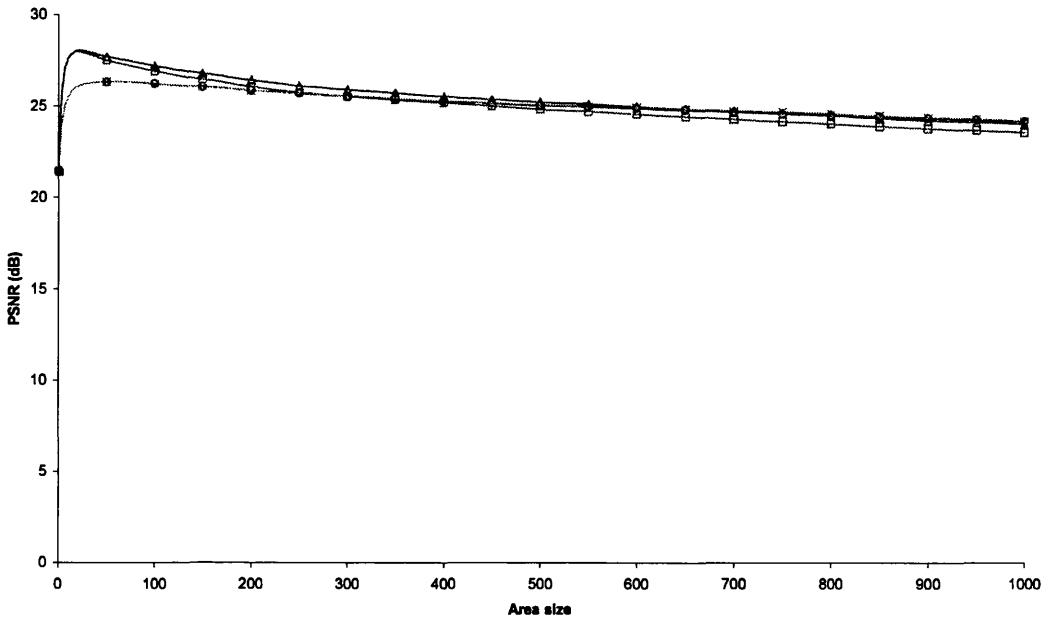
a) Results for the volume attribute.



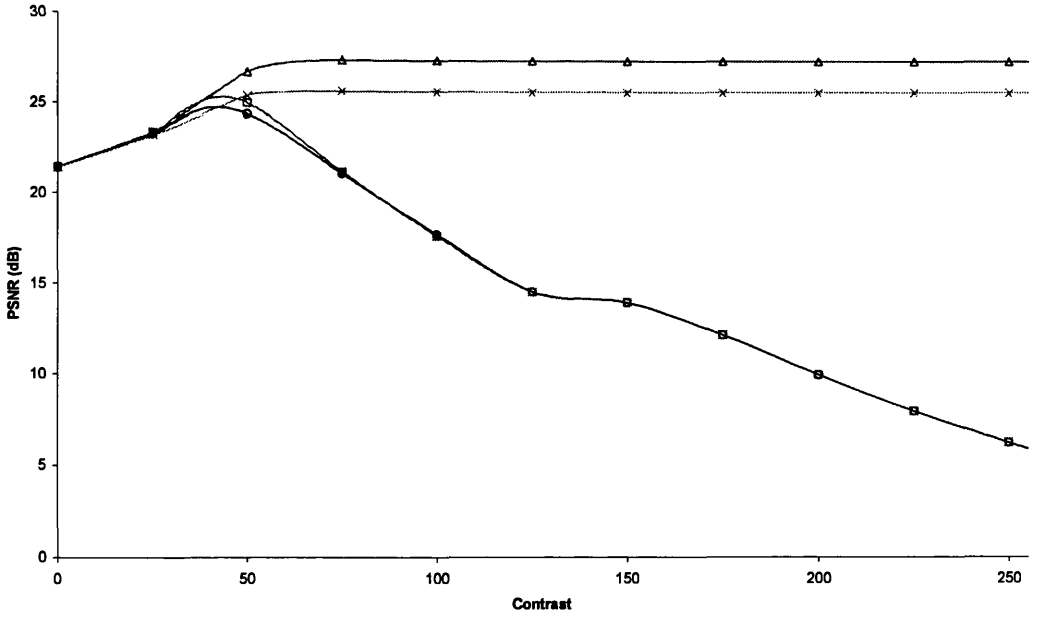
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.22: Volume and power attribute results for the Boats image with 14dB of noise.



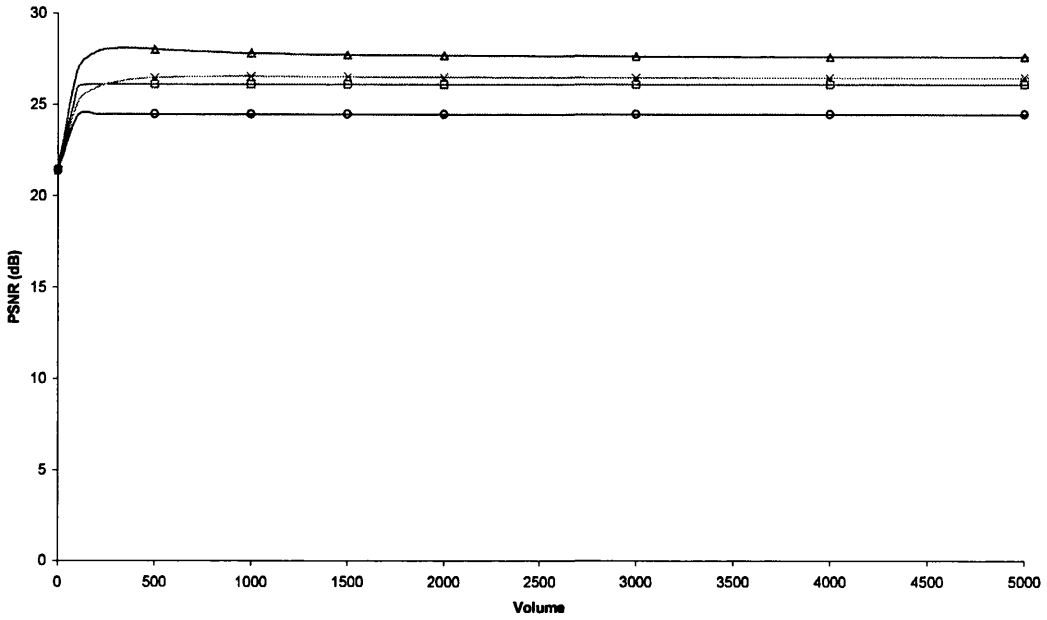
a) Results for the area attribute.



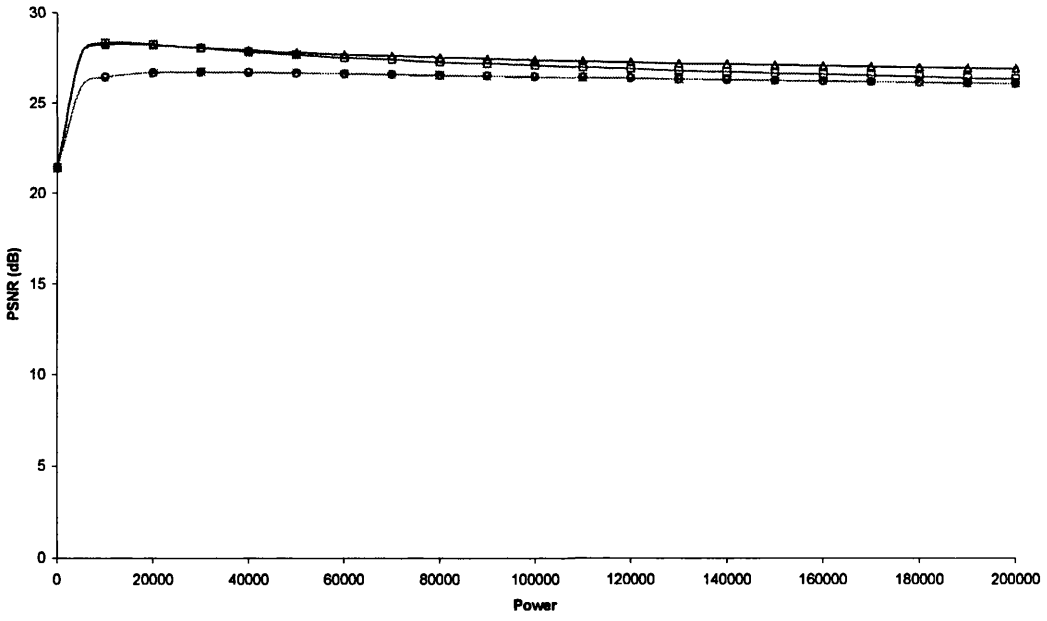
b) Results for the contrast attribute.

Key: 4nnAF 4nnASF 8nnAF 8nnASF

Figure 13.23: Area and contrast attribute results for the Boats image with 21dB of noise.



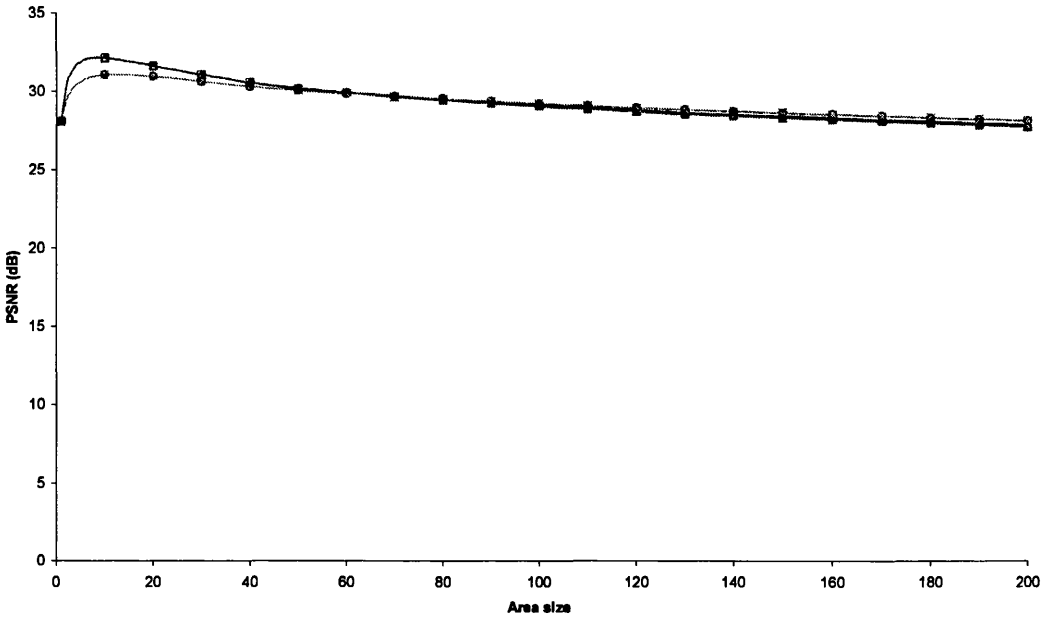
a) Results for the volume attribute.



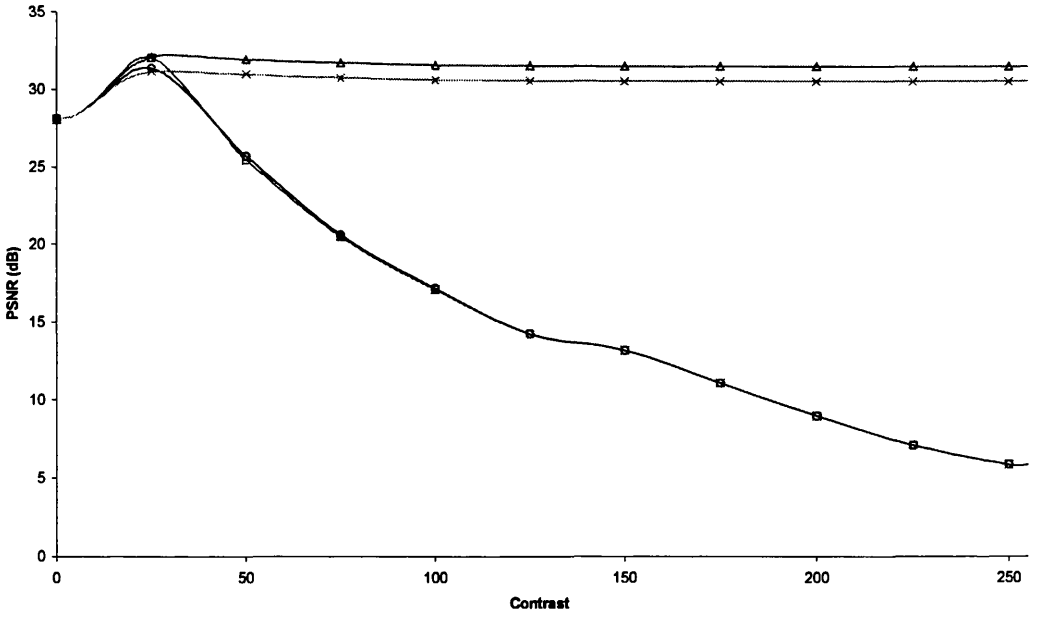
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

igure 13.24: Volume and power attribute results for the Boats image with 21dB of noise.



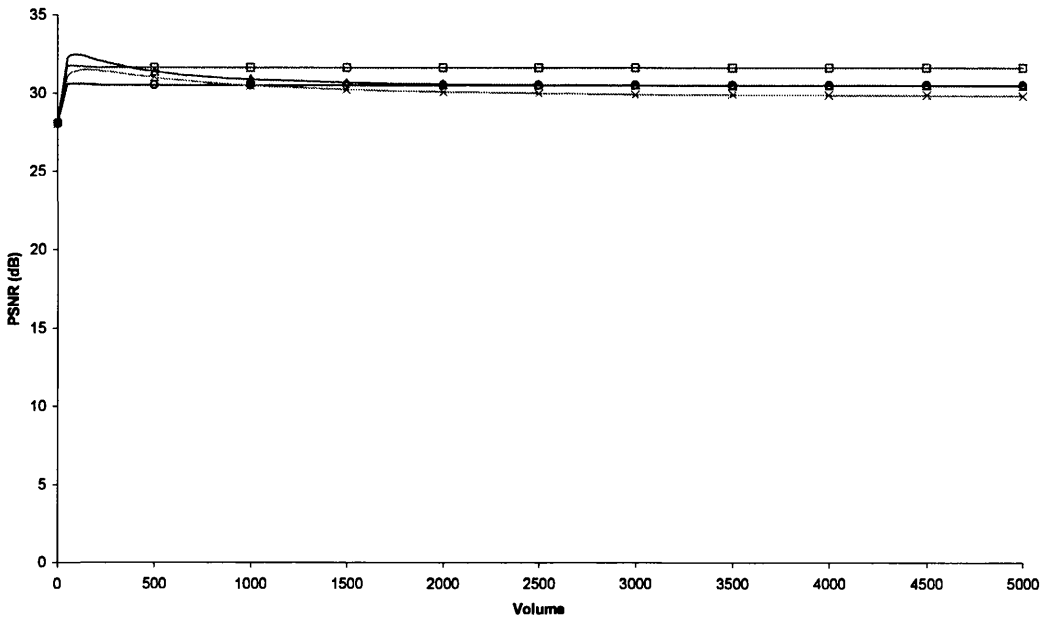
a) Results for the area attribute.



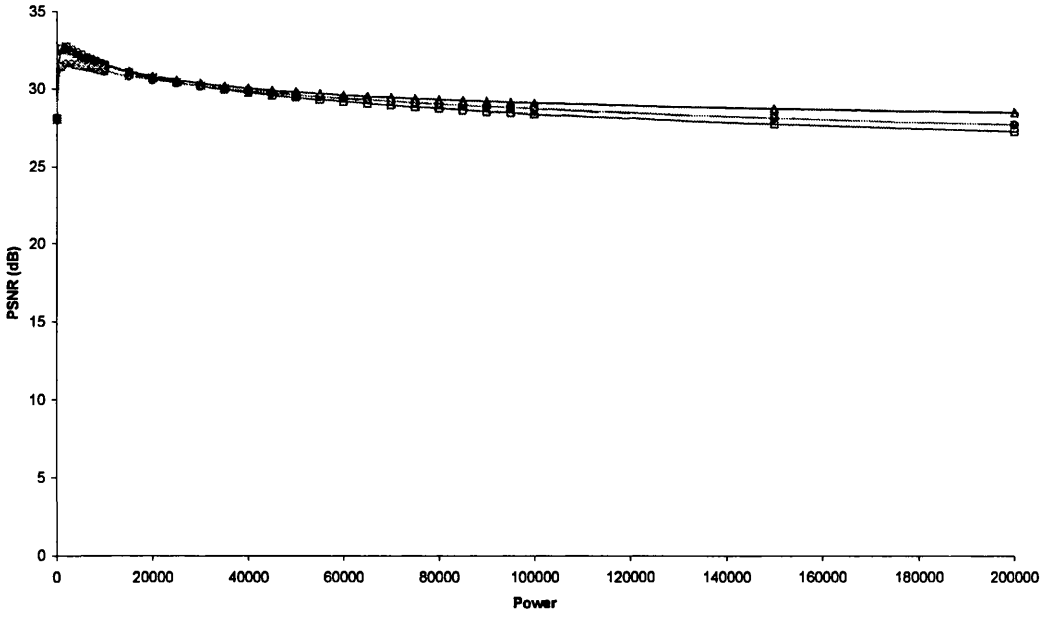
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.25: Area and contrast attribute results for the Boats image with 28dB of noise.



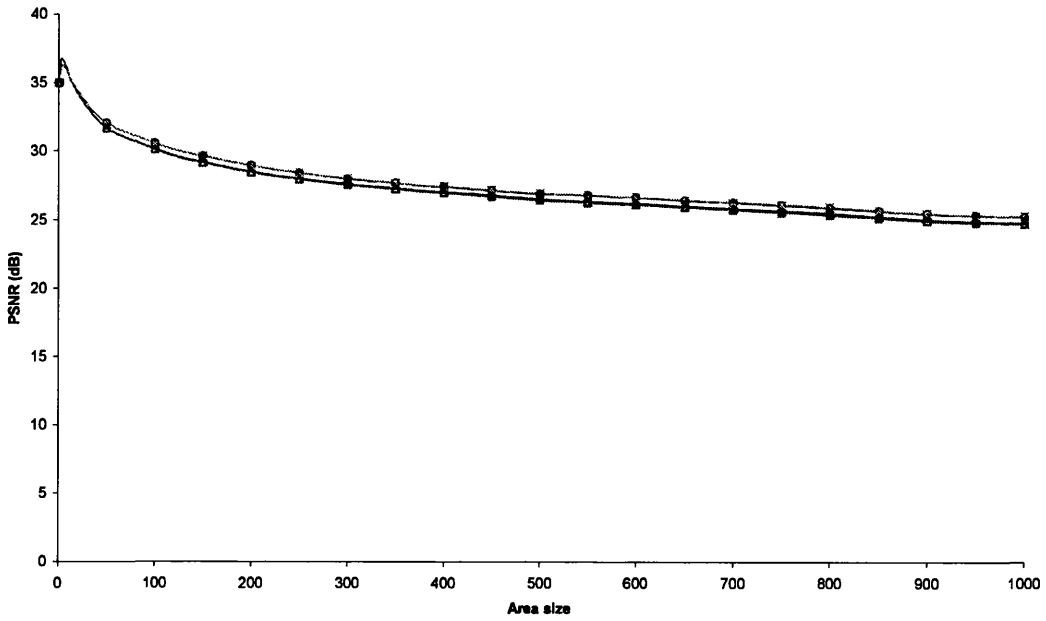
a) Results for the volume attribute.



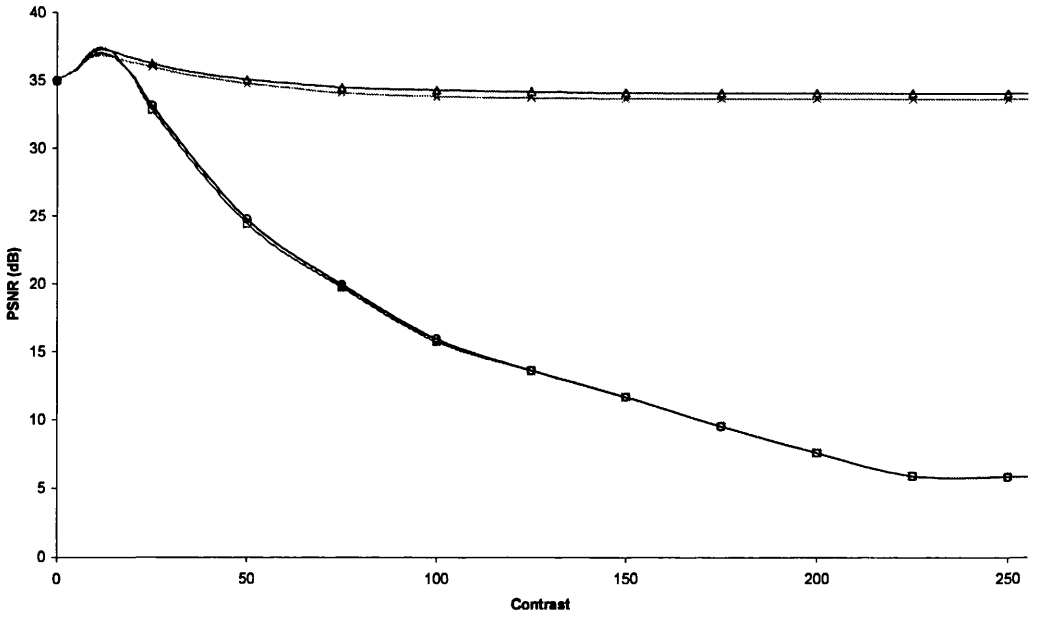
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.26: Volume and power attribute results for the Boats image with 28dB of noise.



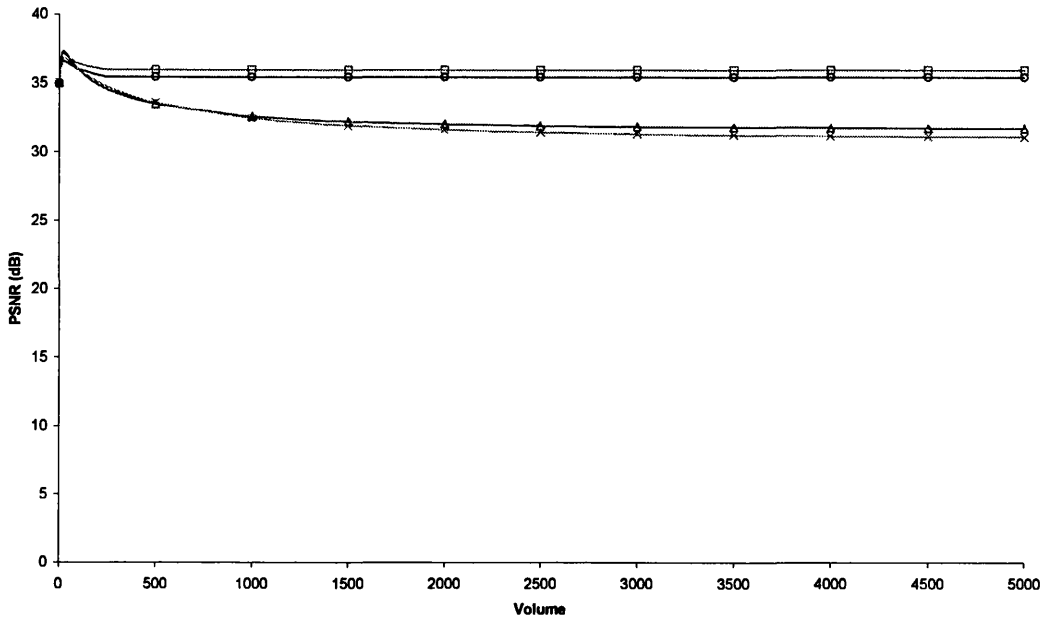
a) Results for the area attribute.



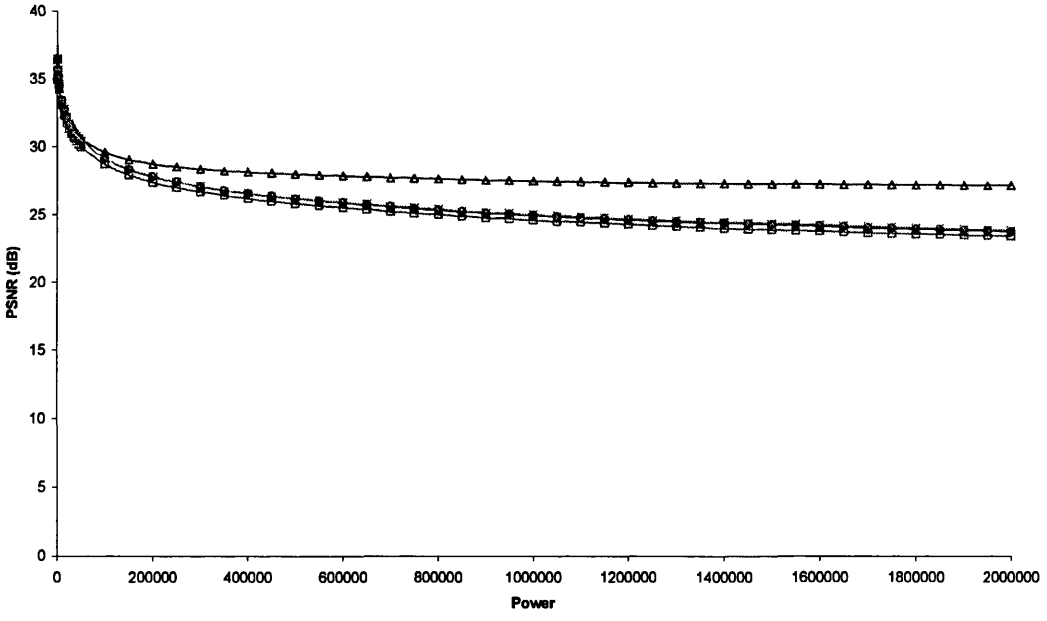
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.27: Area and contrast attribute results for the Boats image with 35dB of noise.



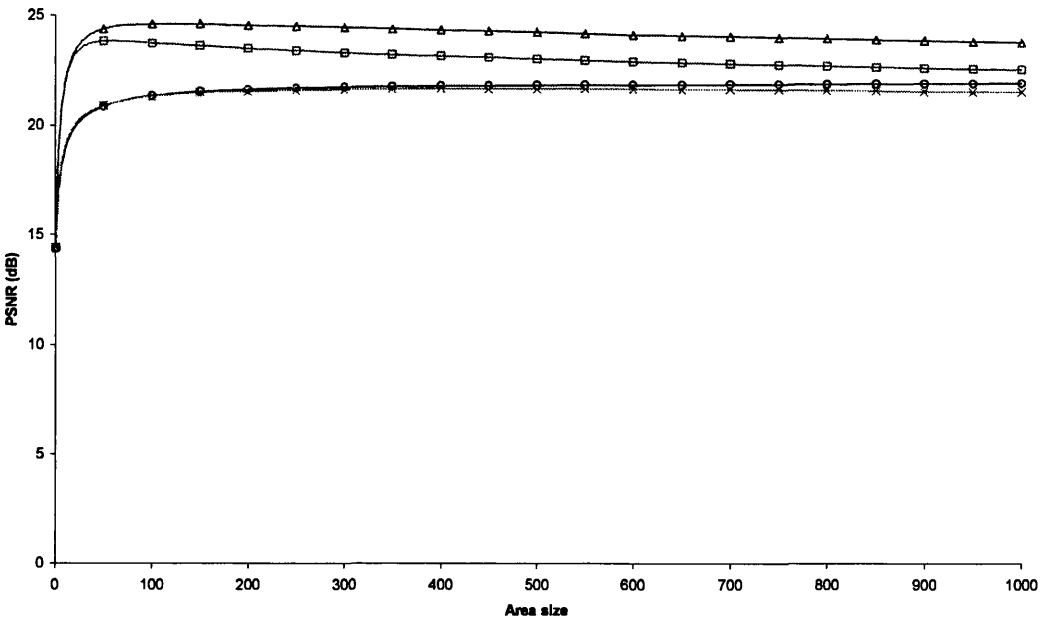
a) Results for the volume attribute.



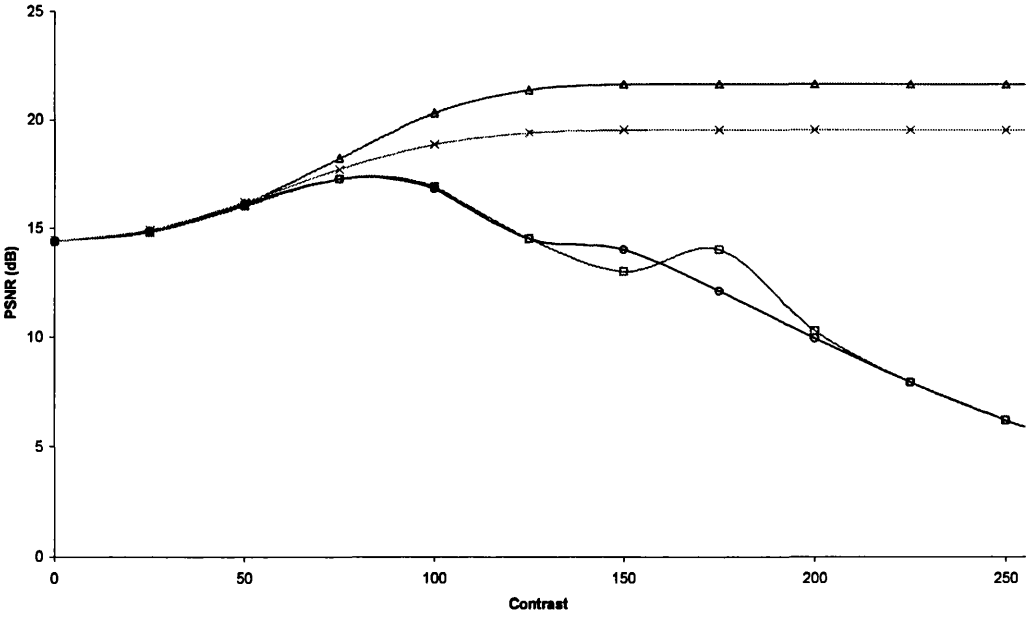
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.28: Volume and power attribute results for the Boats image with 35dB of noise.



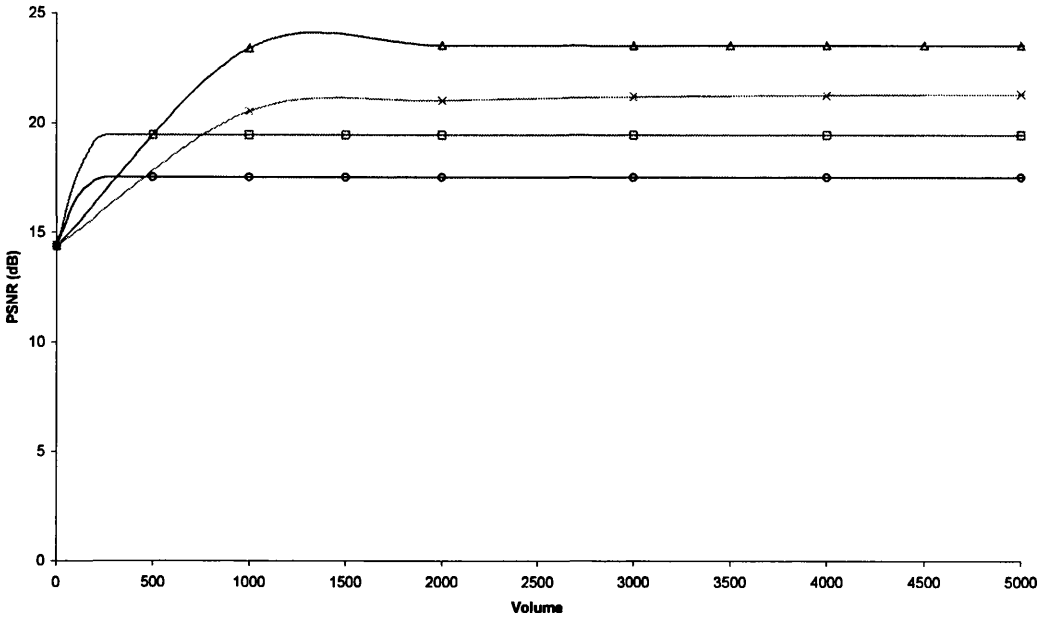
a) Results for the area attribute.



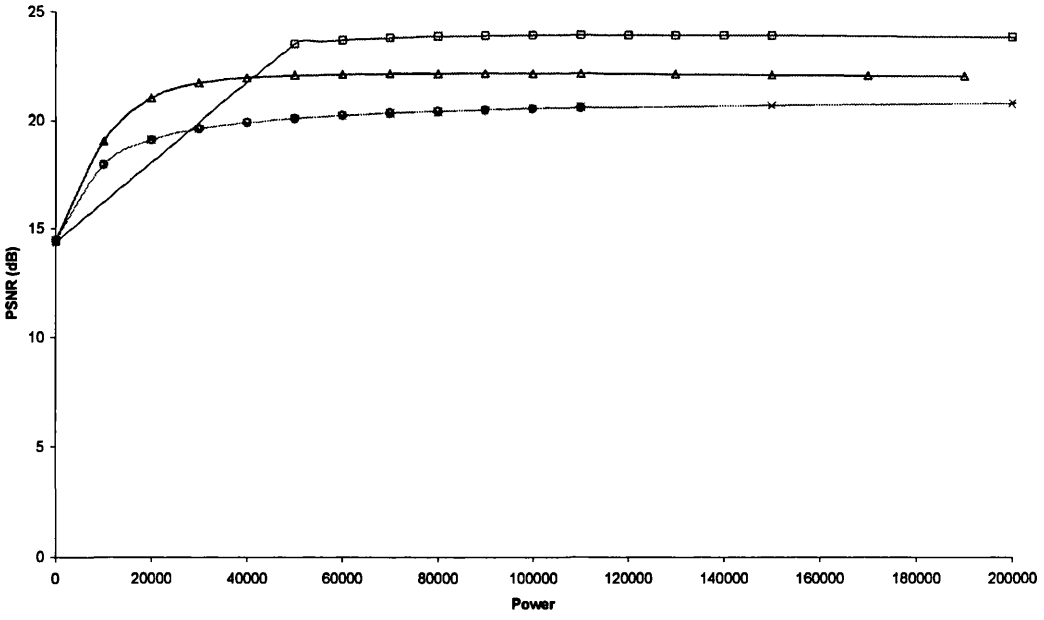
b) Results for the contrast attribute.

Key: 4nmAF 4nmASF 8nmAF 8nmASF

Figure 13.29: Area and contrast attribute results for the Goldhill image with 14dB of noise.



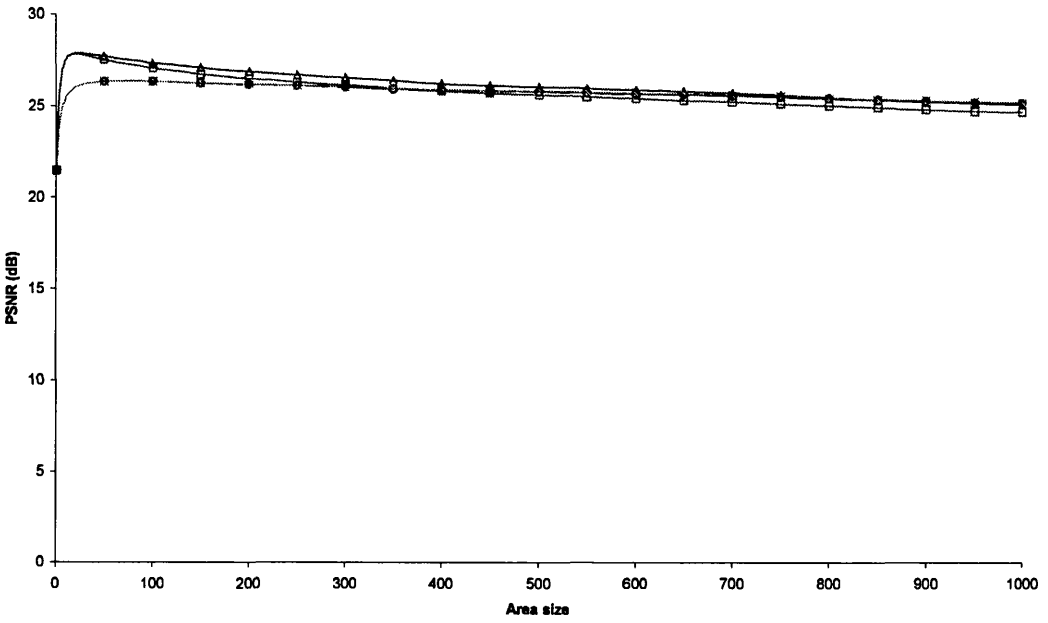
a) Results for the volume attribute.



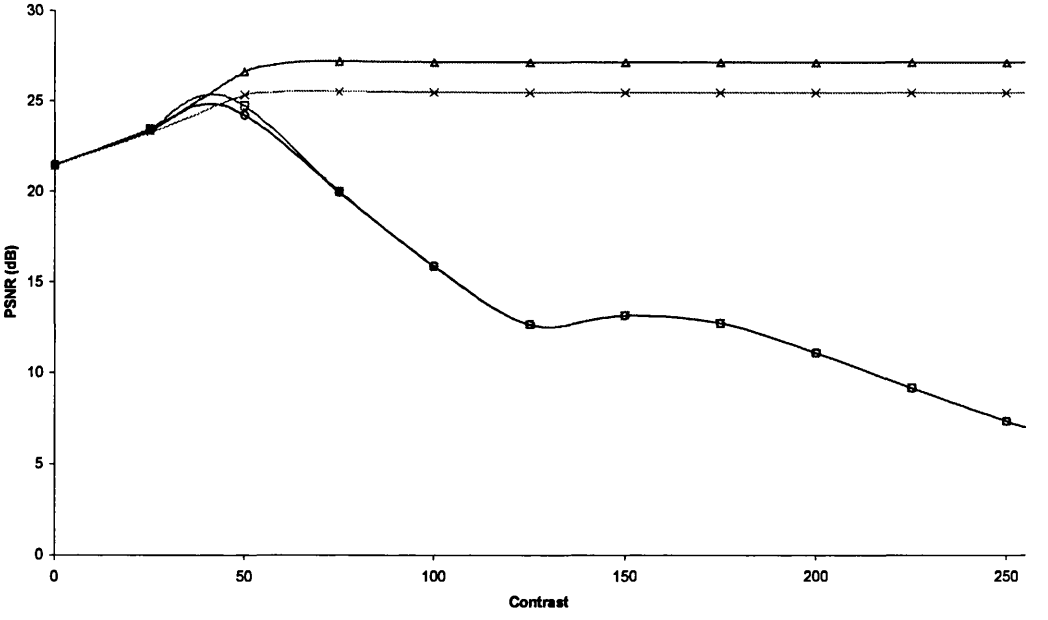
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.30: Volume and power attribute results for the Goldhill image with 14dB of noise.



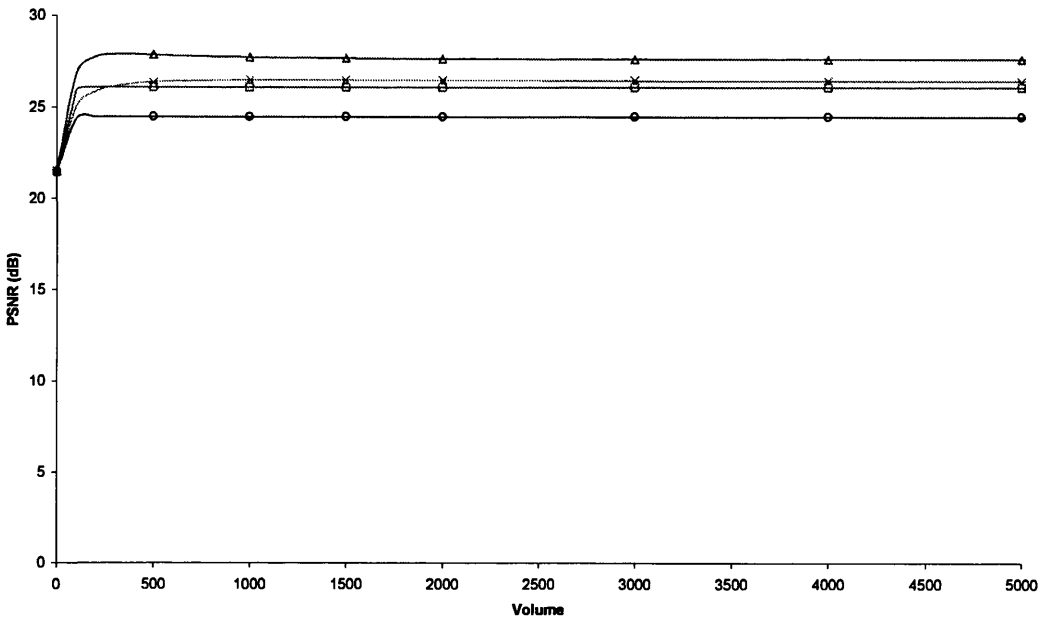
a) Results for the area attribute.



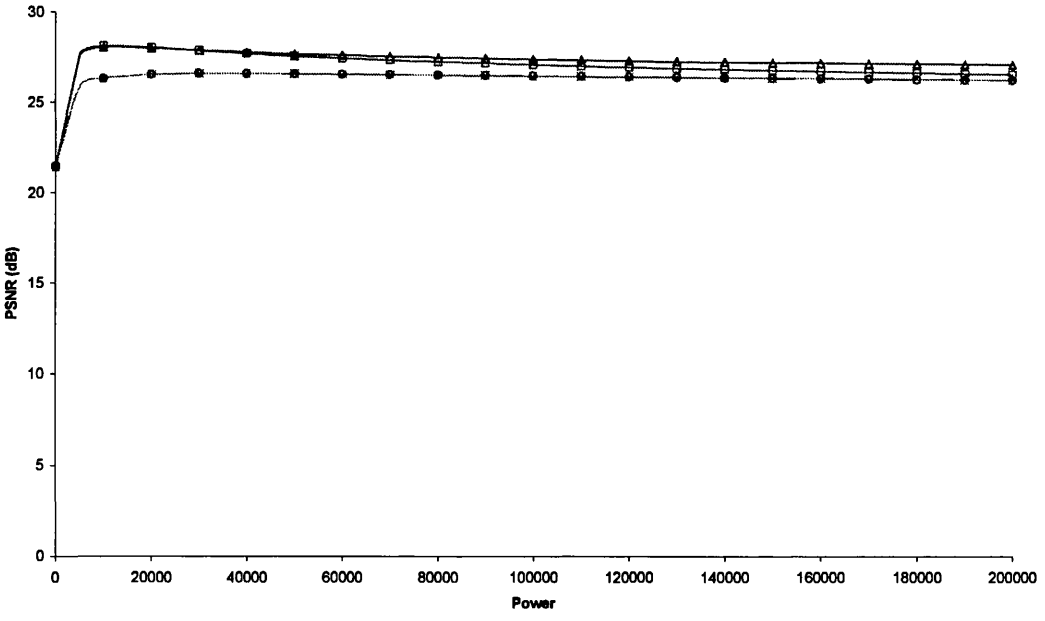
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.31: Area and contrast attribute results for the Goldhill image with 21dB of noise.



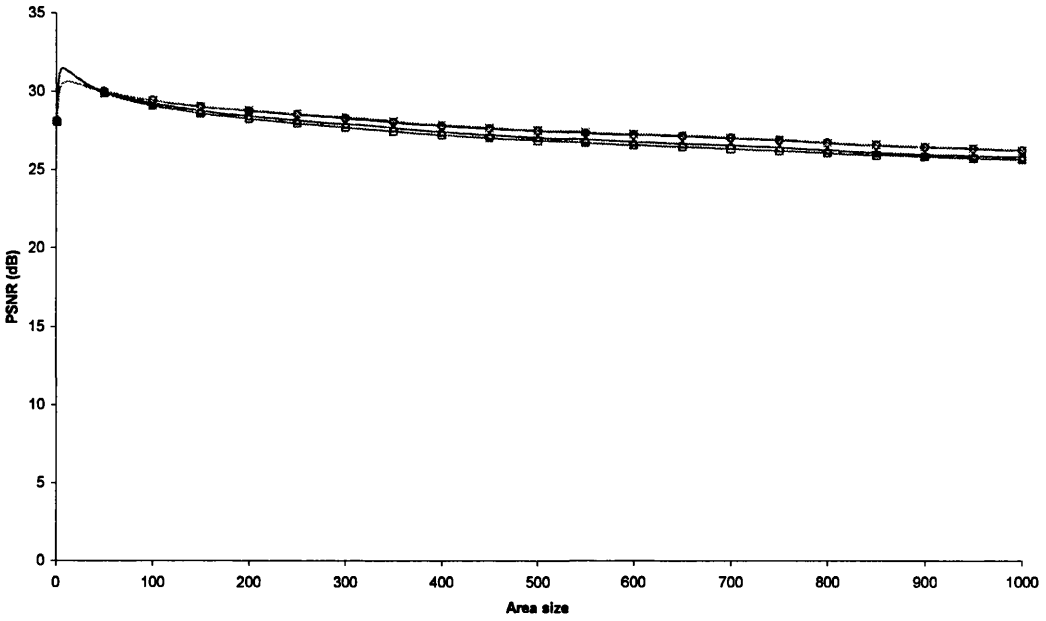
a) Results for the volume attribute.



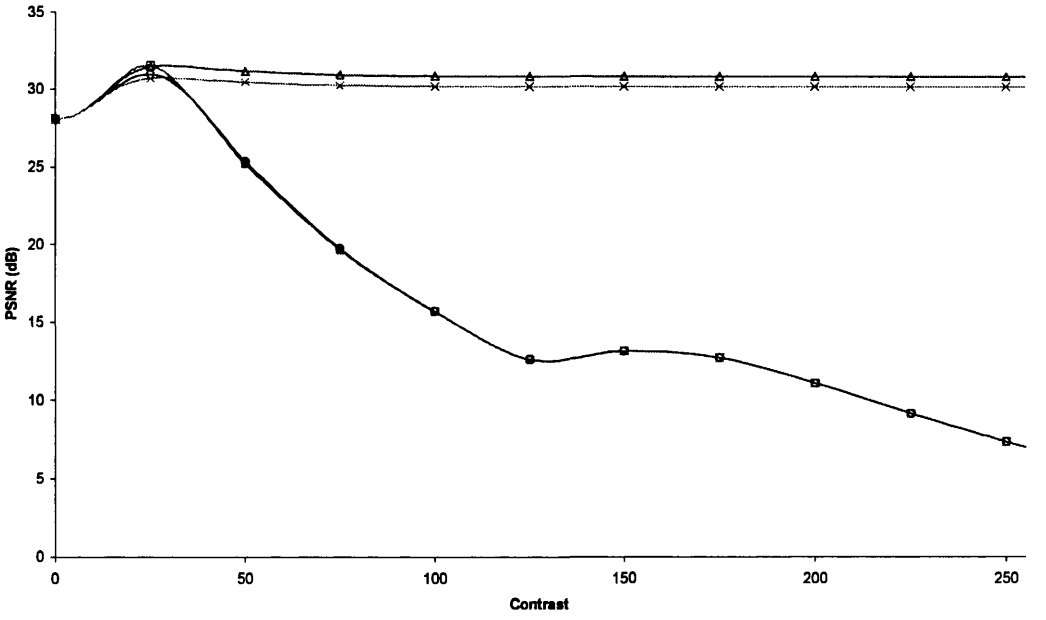
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.32: Volume and power attribute results for the Goldhill image with 21dB of noise.



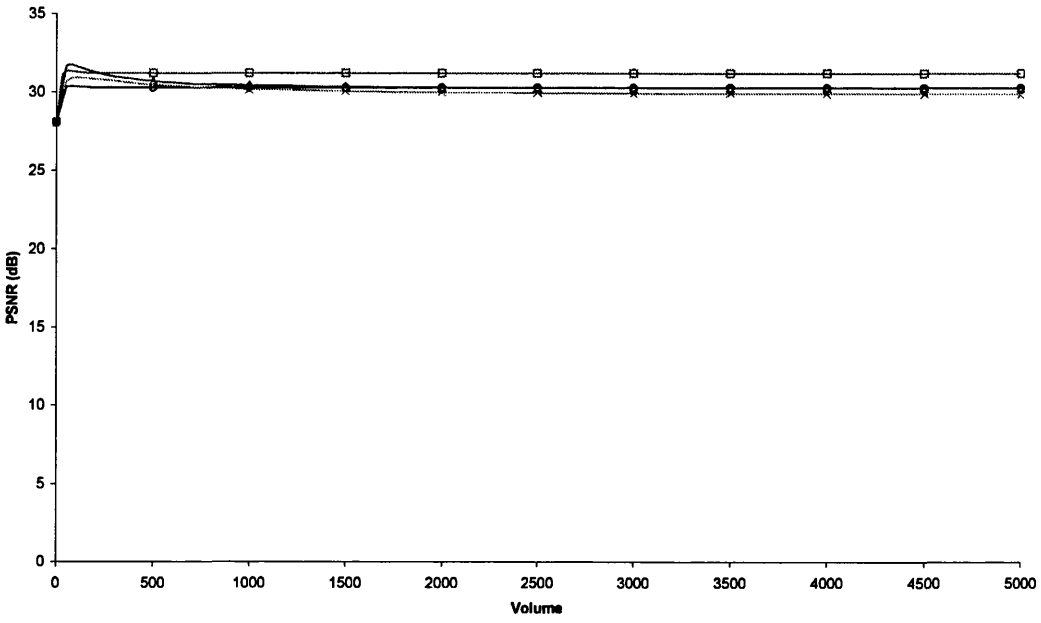
a) Results for the area attribute.



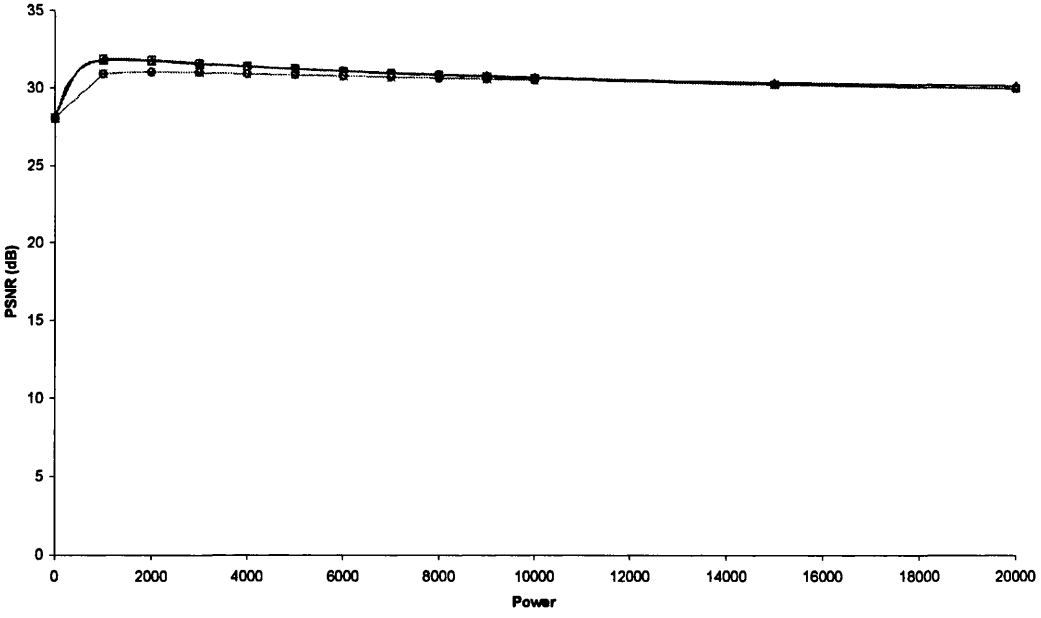
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.33: Area and contrast attribute results for the Goldhill image with 28dB of noise.



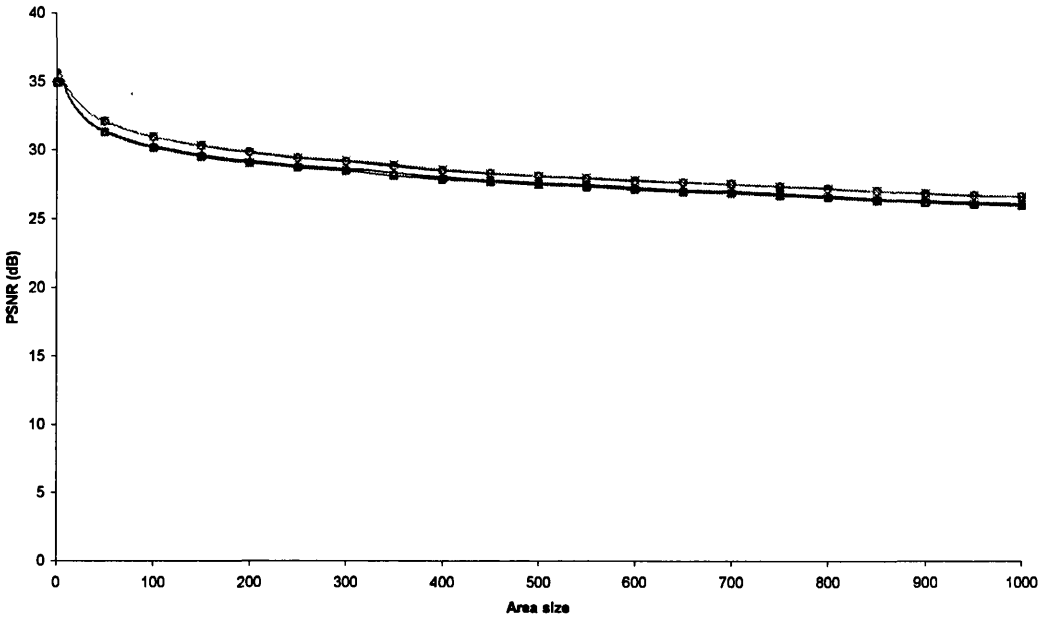
a) Results for the volume attribute.



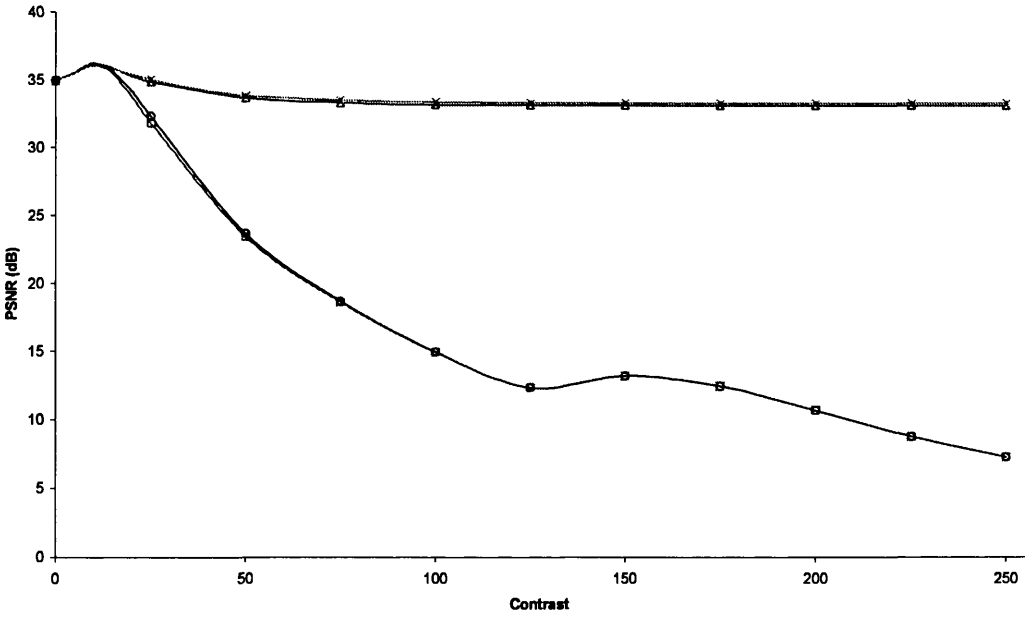
b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.34: Volume and power attribute results for the Goldhill image with 28dB of noise.



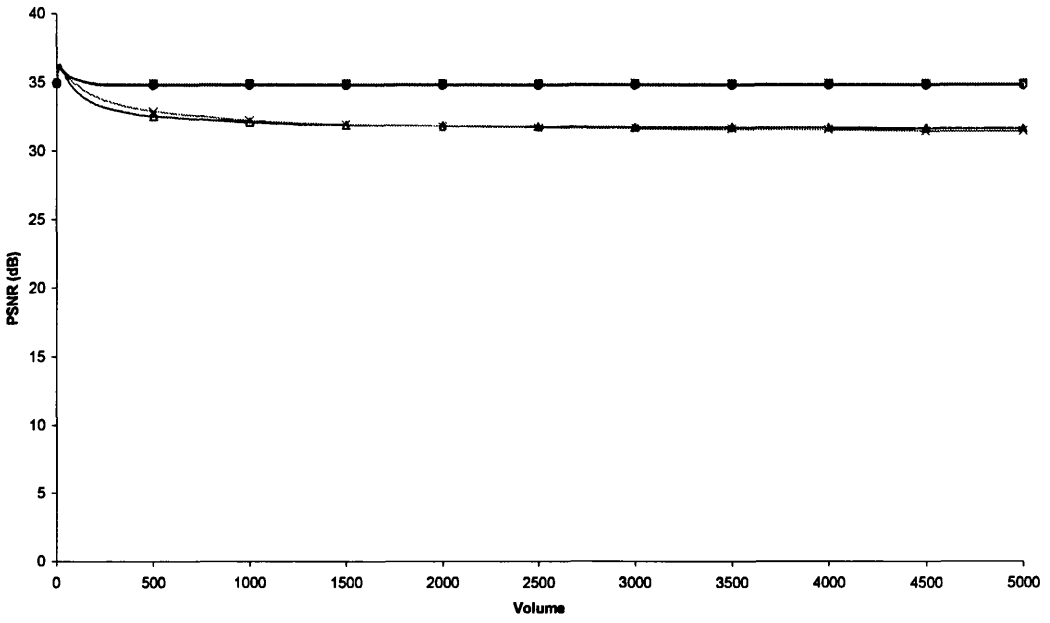
a) Results for the area attribute.



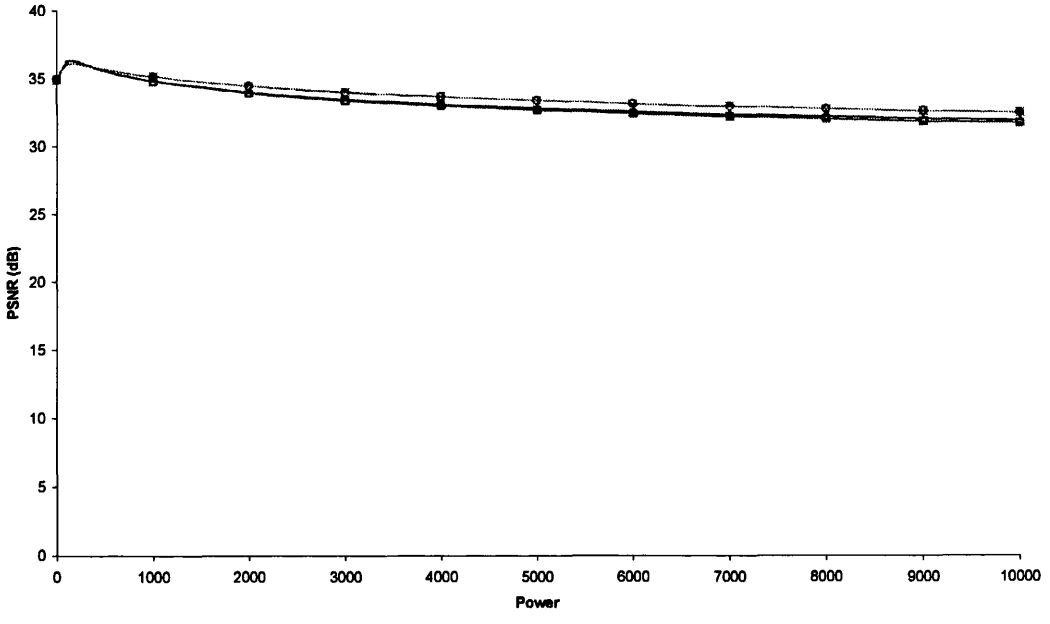
b) Results for the contrast attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.35: Area and contrast attribute results for the Goldhill image with 35dB of noise.



a) Results for the volume attribute.



b) Results for the power attribute.

Key: 4nn AF 4nn ASF 8nn AF 8nn ASF

Figure 13.36: Volume and power attribute results for the Goldhill image with 35dB of noise.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 14.47dB	Area	34	21.97	59	**22.25**	939	20.95	912	20.95
	Contrast	79	17.56	217	20.96	78	17.53	224	19.22
	Volume	256	19.19	1108	21.91	256	17.47	8862	20.63
	Power	71906	22.21	91051	22.19	111066	20.62	912139	21.04
Boats 14.41dB	Area	44	23.56	88	**24.12**	469	21.68	459	21.69
	Contrast	84	17.43	191	22.00	82	17.39	240	19.45
	Volume	256	19.52	1620	23.34	256	17.61	10899	21.26
	Power	71906	23.78	159316	23.82	889318	21.80	889902	21.79
Goldhill 14.39dB	Area	44	23.85	136	**24.63**	1000	21.94	219	19.60
	Contrast	84	16.95	206	22.00	71	17.01	219	19.60
	Volume	256	19.45	3861	23.54	256	17.52	10956	21.36
	Power	103616	23.96	202127	24.11	2532300	22.05	2533680	22.05
Barbara 2 21.49dB	Area	9	25.85	9	25.85	19	24.91	19	24.91
	Contrast	44	24.84	61	25.69	43	24.37	59	24.73
	Volume	139	25.31	176	26.16	184	24.16	336	25.19
	Power	6632	**26.36**	6632	26.32	10667	25.31	10667	25.31
Boats 21.43dB	Area	20	25.85	22	28.06	63	26.36	63	26.36
	Contrast	43	24.84	72	27.33	42	24.69	68	25.59
	Volume	217	25.31	356	28.10	255	24.50	952	26.59
	Power	11745	26.36	13079	**28.29**	31668	26.73	31668	26.73
Goldhill 21.47dB	Area	22	27.84	23	27.90	85	26.37	85	26.37
	Contrast	41	25.37	71	27.22	41	24.84	66	25.54
	Volume	184	26.11	338	27.91	255	24.49	1203	26.50
	Power	11400	**28.15**	11744	28.07	39377	26.60	37498	26.60

Table 13.3: The optimum attribute values for noise removal and their performance gain in PSNR for noise levels of 14dB and 21dB. Best results are shaded and the worst are outlined.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 28.08dB	Area	4	30.22	4	30.22	5	29.78	5	29.78
	Contrast	22	30.66	25	30.57	23	30.31	25	30.13
	Volume	42	30.55	47	30.74	50	29.92	64	30.20
	Power	902	**30.90**	902	30.87	1130	30.34	1130	30.34
Boats 28.11dB	Area	8	32.19	8	32.16	12	31.09	12	31.09
	Contrast	24	32.10	29	32.23	23	31.42	28	31.19
	Volume	66	31.78	88	32.51	74	30.65	149	31.52
	Power	1533	**32.76**	1518	32.67	2355	31.67	2355	31.67
Goldhill 28.09dB	Area	7	31.49	7	31.48	12	30.65	12	30.65
	Contrast	24	31.53	28	31.55	25	30.98	28	30.74
	Volume	58	31.35	75	31.74	78	30.37	116	30.94
	Power	1298	**31.92**	1299	31.86	1910	31.05	1910	31.05
Barbara 2 34.97dB	Area	2	35.50	2	35.50	2	35.46	2	35.46
	Contrast	10	36.19	10	36.10	10	36.07	10	36.00
	Volume	13	36.08	13	36.08	14	35.88	15	35.91
	Power	125	**36.20**	125	36.17	145	36.04	145	36.04
Boats 34.99dB	Area	3	36.79	3	36.78	4	36.29	4	36.29
	Contrast	11	37.44	12	37.29	12	37.04	12	36.86
	Volume	20	37.21	22	37.33	21	36.63	26	36.82
	Power	221	**37.55**	202	37.48	241	37.00	241	37.00
Goldhill 34.96dB	Area	2	35.92	2	35.92	3	35.68	3	35.68
	Contrast	10	36.25	10	36.20	10	36.08	10	36.06
	Volume	15	36.30	15	36.31	17	35.99	18	36.03
	Power	145	**36.37**	145	36.34	161	36.11	161	36.11

Table 13.4: The optimum attribute values for noise removal and their performance gain in PSNR for noise levels of 28dB and 35dB. Best results are shaded and the worst are outlined.

13.1.3 Compression Results

This section contains the full results from section 7.1.2.1.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 (Original noise free)			27.66						
Boats (Original noise free)			32.66						
Goldhill (Original noise free)			31.39						
Barbara 2 14.47dB	Noisy		18.64						
	Area	34	22.05	59	22.31	939	21.79	912	21.80
	Contrast	79	18.71	217	21.50	78	18.79	224	21.30
	Volume	256	20.35	1108	20.35	256	20.22	8862	20.22
	Power	71906	22.31	91051	**22.36**	111066	22.19	912139	22.02
Boats 14.41dB	Noisy		18.58						
	Area	44	23.43	88	**24.48**	469	23.41	459	23.43
	Contrast	84	18.38	191	22.72	82	18.72	240	22.22
	Volume	256	21.10	1620	21.10	256	21.02	10899	21.02
	Power	103616	24.21	159316	24.42	889318	23.56	889902	23.53
Goldhill 14.39dB	Noisy		18.51						
	Area	59	23.66	136	**24.93**	1000	23.69	219	23.90
	Contrast	72	18.29	206	23.02	71	18.59	219	22.48
	Volume	256	21.05	3861	21.05	256	20.96	10956	21.02
	Power	131619	24.32	202127	24.68	25323000	23.75	2533680	23.75
Barbara 2 21.49dB	Noisy		23.95						
	Area	9	25.24	9	25.25	19	25.14	19	25.14
	Contrast	44	24.72	61	25.18	43	24.77	59	25.20
	Volume	139	24.98	176	24.96	184	25.06	336	25.04
	Power	6632	**25.58**	6632	25.57	10667	25.48	10667	25.48
Boats 21.43dB	Noisy		25.41						
	Area	20	28.16	22	28.24	63	27.61	63	27.62
	Contrast	43	26.21	72	27.79	42	26.27	68	27.59
	Volume	217	27.04	356	27.04	255	26.96	952	26.96
	Power	11745	**28.51**	13079	**28.51**	31668	28.09	31668	28.09
Goldhill 21.47dB	Noisy		25.11						
	Area	22	27.90	23	27.98	85	27.49	85	27.49
	Contrast	41	26.36	71	27.64	41	26.30	66	27.41
	Volume	184	26.92	338	26.91	255	26.82	1203	26.82
	Power	11400	**28.20**	11744	28.19	39377	27.78	37498	27.79

Table 13.5: Output PSNR of the optimum attribute values using JPEG for 14dB and 21dB of noise.

Best results are shaded and the worst are outlined.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 (Original noise free)			27.66						
Boats (Original noise free)			32.66						
Goldhill (Original noise free)			31.39						
Barbara 2 28.08dB	Noisy		26.78						
	Area	4	27.03	4	27.03	5	27.13	5	27.13
	Contrast	22	27.36	25	27.32	23	27.37	25	27.20
	Volume	42	27.19	47	27.18	50	27.23	46	27.20
	Power	902	27.39	902	27.38	1130	**27.40**	1130	**27.40**
Boats 28.11dB	Noisy		29.98						
	Area	8	31.02	8	31.01	12	30.97	12	30.97
	Contrast	24	31.03	29	31.11	23	31.04	28	31.07
	Volume	66	30.87	88	30.84	74	30.90	149	30.83
	Power	1533	**31.35**	1518	31.32	2355	31.30	2355	31.30
Goldhill 28.09dB	Noisy		29.42						
	Area	7	30.18	7	30.17	12	30.19	12	30.19
	Contrast	24	30.34	28	30.27	25	30.21	28	30.31
	Volume	58	30.10	75	30.15	78	30.10	116	30.08
	Power	1298	**30.42**	1299	30.40	1910	30.36	1910	30.36
Barbara 2 34.97dB	Noisy		27.53						
	Area	2	27.71	2	27.49	2	27.73	2	27.73
	Contrast	10	27.73	10	27.73	10	**27.74**	10	**27.74**
	Volume	13	**27.74**	13	**27.74**	14	**27.74**	15	**27.74**
	Power	125	27.73	125	27.73	145	27.73	145	27.73
Boats 34.99dB	Noisy		32.06						
	Area	3	32.21	3	32.21	4	32.33	4	32.33
	Contrast	11	32.38	12	32.36	12	32.33	12	32.37
	Volume	20	**32.39**	22	32.38	21	32.25	26	32.37
	Power	221	32.38	202	32.37	241	32.38	241	32.38
Goldhill 34.96dB	Noisy		30.89						
	Area	2	31.16	2	31.16	3	31.15	3	31.15
	Contrast	10	**31.28**	10	31.16	10	31.16	10	31.18
	Volume	15	31.17	15	31.17	17	31.18	18	31.17
	Power	145	31.16	145	31.15	161	31.17	161	31.17

Table 13.6: Output PSNR of the optimum attribute values using JPEG for 28dB and 35dB of noise.

Best results are shaded and the worst are outlined.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 (Original noise free)			30.64						
Boats (Original noise free)			35.00						
Goldhill (Original noise free)			32.83						
Barbara 2 14.47dB	Noisy		17.48						
	Area	34	22.02	59	22.25	939	21.44	912	21.44
	Contrast	79	17.99	217	21.81	78	18.24	224	18.97
	Volume	256	20.74	1108	20.74	256	20.35	8862	20.35
	Power	71906	**22.32**	91051	22.25	111066	21.73	912139	21.65
Boats 14.41dB	Noisy		17.87						
	Area	44	22.74	88	**24.27**	469	22.88	459	22.90
	Contrast	84	18.00	191	23.15	82	18.23	240	21.71
	Volume	256	21.53	1620	21.53	256	21.05	10899	21.05
	Power	103616	24.15	159316	24.21	889318	23.01	889902	22.98
Goldhill 14.39dB	Noisy		17.73						
	Area	59	23.16	136	**24.84**	1000	23.30	219	23.52
	Contrast	72	17.74	206	23.41	71	17.96	219	21.78
	Volume	256	21.50	3861	21.50	256	21.00	10956	21.00
	Power	131619	24.35	202127	24.49	25323000	23.32	2533680	23.32
Barbara 2 21.49dB	Noisy		24.42						
	Area	9	25.86	9	25.87	19	25.46	19	25.46
	Contrast	44	24.77	61	25.73	43	25.02	59	25.61
	Volume	139	25.82	176	25.79	184	25.83	336	25.80
	Power	6632	**26.20**	6632	26.16	10667	25.82	10667	25.82
Boats 21.43dB	Noisy		24.33						
	Area	20	28.25	22	28.33	63	27.10	63	27.10
	Contrast	43	25.65	72	28.05	42	25.77	68	27.27
	Volume	217	27.50	356	27.49	255	27.04	952	27.04
	Power	11745	**28.62**	13079	28.55	31668	27.53	31668	27.53
Goldhill 21.47dB	Noisy		24.22						
	Area	22	27.98	23	28.03	85	26.90	85	26.90
	Contrast	41	25.69	71	27.82	41	25.63	66	26.93
	Volume	184	27.30	338	27.29	255	26.74	1203	26.74
	Power	11400	**28.28**	11744	28.23	39377	27.17	37498	27.22

Table 13.7: Output PSNR of the optimum attribute values using JPEG 2000 for 14dB and 21dB of noise.

Image (Noise Level (dB))	Attribute Type	4nn				8nn			
		AF		ASF		AF		ASF	
		Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)	Attribute Value	PSNR (dB)
Barbara 2 (Original noise free)			30.64						
Boats (Original noise free)			35.00						
Goldhill (Original noise free)			32.83						
Barbara 2 28.08dB	Noisy		28.65						
	Area	4	28.88	4	28.88	5	28.91	5	28.91
	Contrast	22	28.94	25	28.93	23	28.92	25	28.98
	Volume	42	29.04	47	29.04	50	29.03	46	29.00
	Power	902	**29.08**	902	29.05	1130	29.05	1130	29.05
Boats 28.11dB	Noisy		30.89						
	Area	8	31.99	8	31.96	12	31.75	12	31.75
	Contrast	24	31.80	29	32.13	23	31.81	28	32.10
	Volume	66	32.12	88	32.08	74	31.95	149	31.86
	Power	1533	**32.30**	1518	32.18	2355	32.02	2355	32.01
Goldhill 28.09dB	Noisy		29.92						
	Area	7	30.85	7	30.84	12	30.61	12	30.62
	Contrast	24	30.74	28	30.93	25	30.62	28	30.86
	Volume	58	30.96	75	30.93	78	30.84	116	30.81
	Power	1298	**31.05**	1299	31.00	1910	30.86	1910	30.86
Barbara 2 34.97dB	Noisy		30.13						
	Area	2	30.24	2	30.24	2	30.22	2	30.22
	Contrast	10	30.25	10	30.23	10	30.19	10	30.21
	Volume	13	30.23	13	30.22	14	30.22	15	30.19
	Power	125	**30.26**	125	30.21	145	30.21	145	30.21
Boats 34.99dB	Noisy		34.17						
	Area	3	34.29	3	34.29	4	34.26	4	34.26
	Contrast	11	**34.44**	12	34.38	12	34.31	12	34.39
	Volume	20	34.39	22	34.40	21	34.37	26	34.35
	Power	221	34.43	202	34.37	241	34.36	241	34.36
Goldhill 34.96dB	Noisy		32.36						
	Area	2	32.46	2	32.46	3	32.47	3	32.47
	Contrast	10	32.50	10	32.50	10	32.49	10	32.51
	Volume	15	32.53	15	32.53	17	32.52	18	32.52
	Power	145	**32.55**	145	32.52	161	32.51	161	32.51

Table 13.8: Output PSNR of the optimum attribute values using JPEG 2000 for 28dB and 35dB of noise.

13.1.4 Application to unknown data

This section contains all of the quadratic equations corresponding to a LMS fit of the data obtained in section 7.1.3. Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the AF filter structure and the contrast attribute:

$$y = \left| -0.0000211x^2 + 0.0793434 + 11.6388422 \right| \tag{13.1}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the AF filter structure and the volume attribute:

$$y = \left| -0.0001211x^2 + 0.3882202x + 9.8743641 \right| \tag{13.2}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the AF filter structure and the power attribute:

$$y = \left| 0.0086563x^2 + 14.0057838x - 44.9765287 \right| \tag{13.3}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the ASF filter structure and the area attribute:

$$y = \left| 0.0000062x^2 + 0.0208992x + 2.8584565 \right| \tag{13.4}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the ASF filter structure and the contrast attribute:

$$y = \left| -0.0000212x^2 + 0.1301373x + 10.3549241 \right| \tag{13.5}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the ASF filter structure and the volume attribute:

$$y = \left| 0.0002509x^2 + 0.2300954x + 41.6305286 \right| \tag{13.6}$$

Estimation of the attribute parameter for the morphological filter using 4nn connectivity, the ASF filter structure and the power attribute:

$$y = \left| 0.0244066x^2 - 1.2094741x + 1592.5474005 \right| \tag{13.7}$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the AF filter structure and the area attribute:

$$y = \left|0.0001537x^2 - 0.0133906x + 8.9570369\right| \quad (13.8)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the AF filter structure and the contrast attribute:

$$y = \left|-0.0000216x^2 + 0.0786207x + 11.8432726\right| \quad (13.9)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the AF filter structure and the volume attribute:

$$y = \left|-0.000176x^2 + 0.516959 + 8.5126231\right| \quad (13.10)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the AF filter structure and the power attribute:

$$y = \left|0.2982518x^2 - 234.3243854x + 26816.6527615\right| \quad (13.11)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the ASF filter structure and the area attribute:

$$y = \left|0.0000448x^2 + 0.113501x - 5.1560241\right| \quad (13.12)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the ASF filter structure and the contrast attribute:

$$y = \left|-0.0000113x^2 + 0.1201915x + 10.6581231\right| \quad (13.13)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the ASF filter structure and the volume attribute:

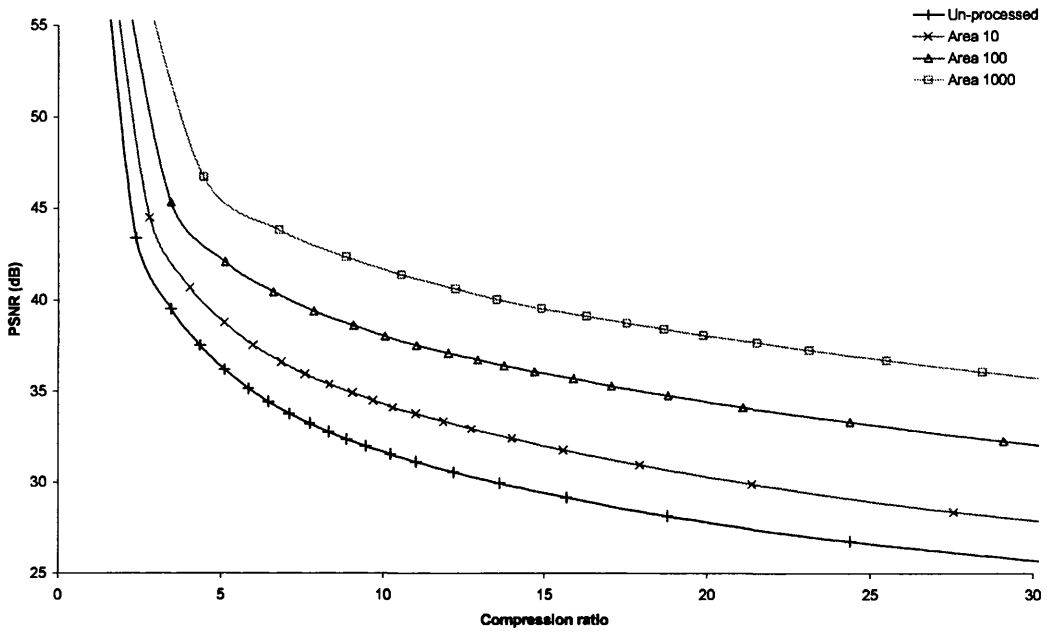
$$y = \left|0.0015502x^2 + 0.5720002x + 33.7344984\right| \quad (13.14)$$

Estimation of the attribute parameter for the morphological filter using 8nn connectivity, the ASF filter structure and the power attribute:

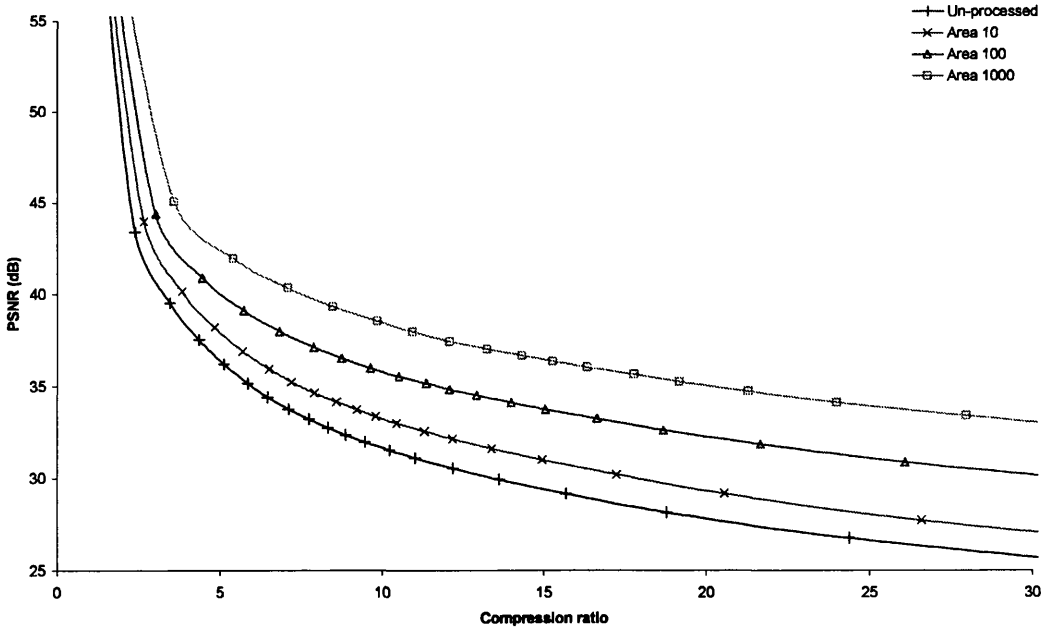
$$y = \left|0.3260906x^2 - 212.1594666x + 21511.3171681\right| \quad (13.15)$$

13.2 Image Simplification

This section contains the full set of results obtained in section 7.2.1.

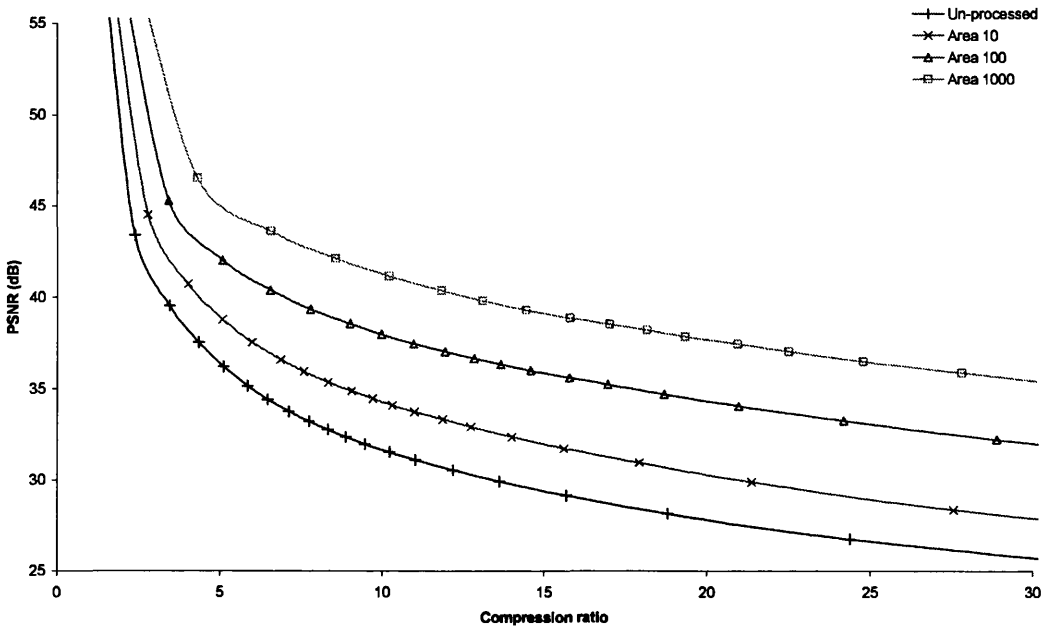


a) 4nn AF

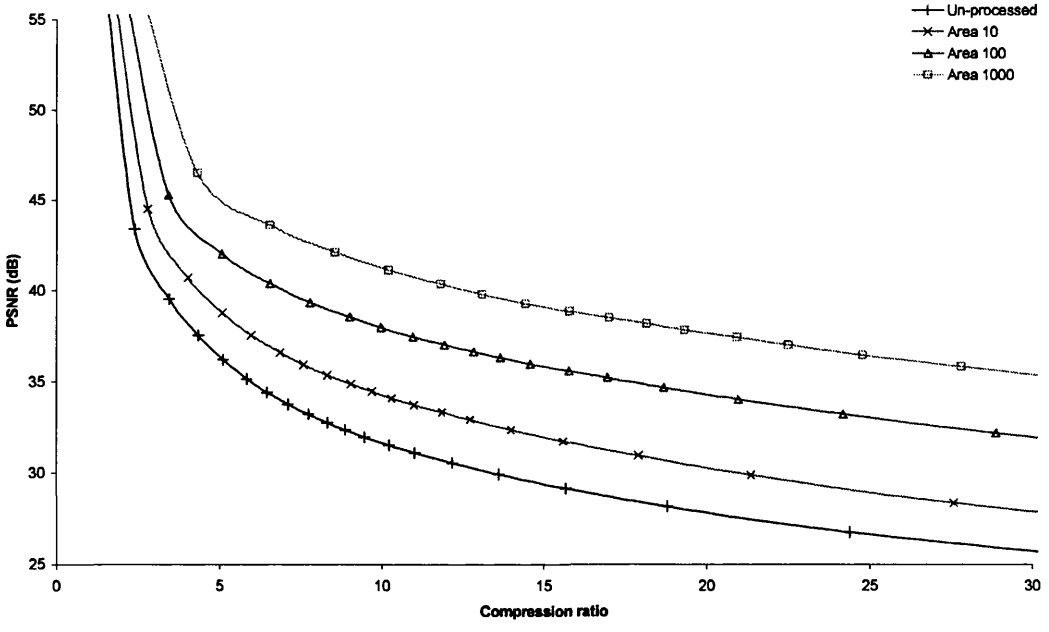


b) 8nn AF

Figure 13.37: JPEG rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the AF filter structure.

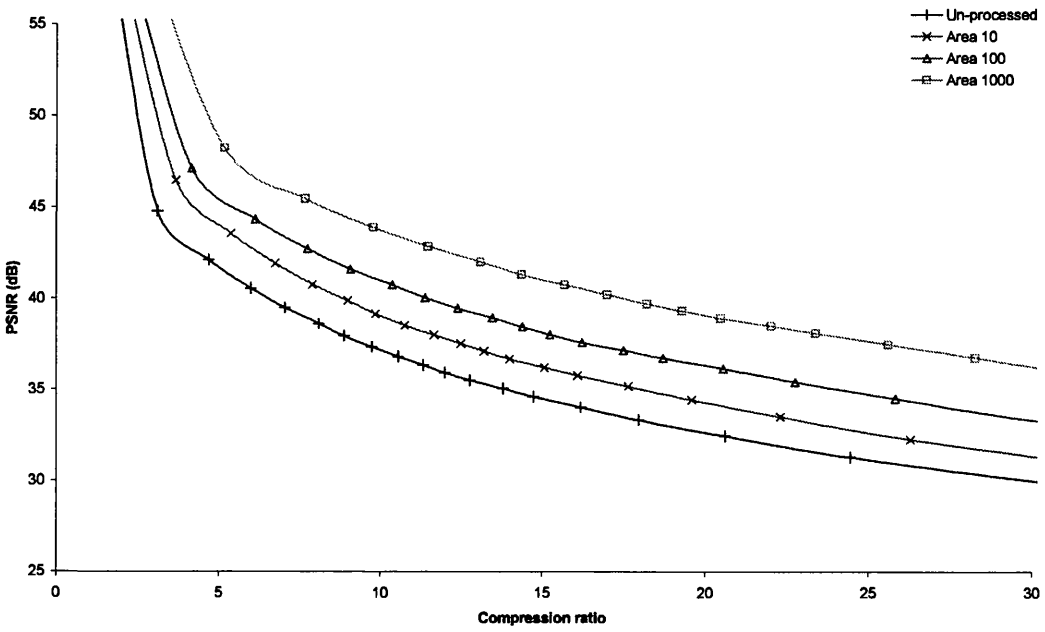


a) 4nn ASF

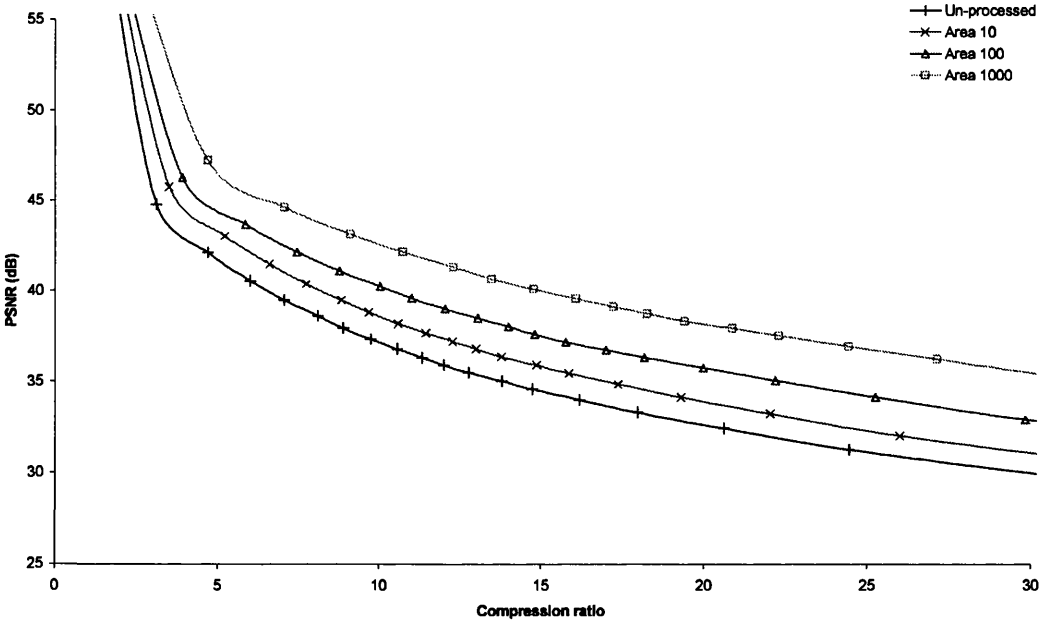


b) 8nn ASF

Figure 13.38: JPEG rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the ASF filter structure.

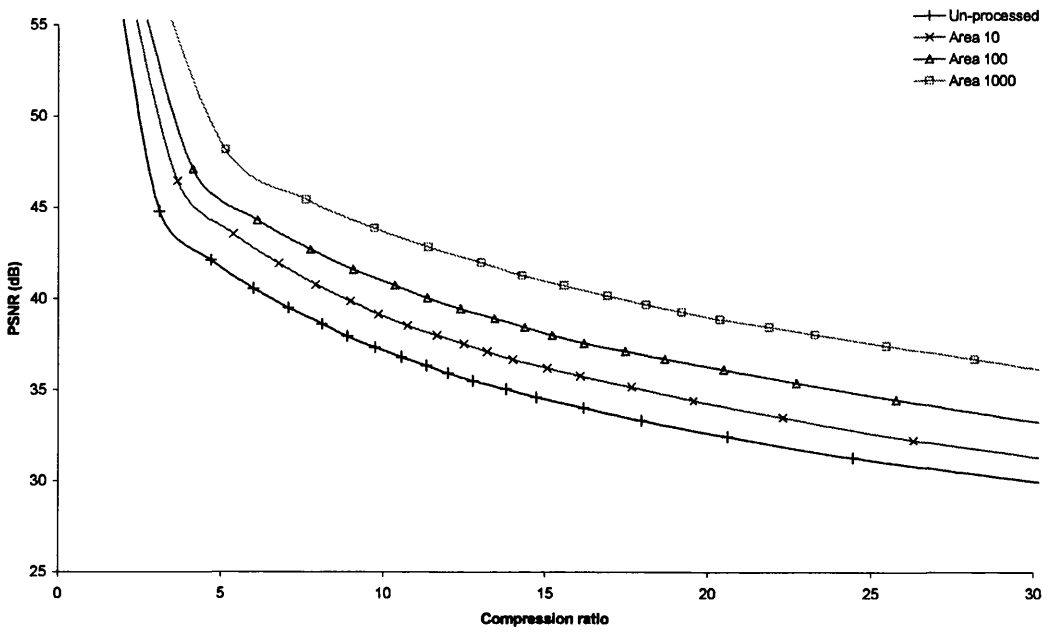


a) 4nn AF

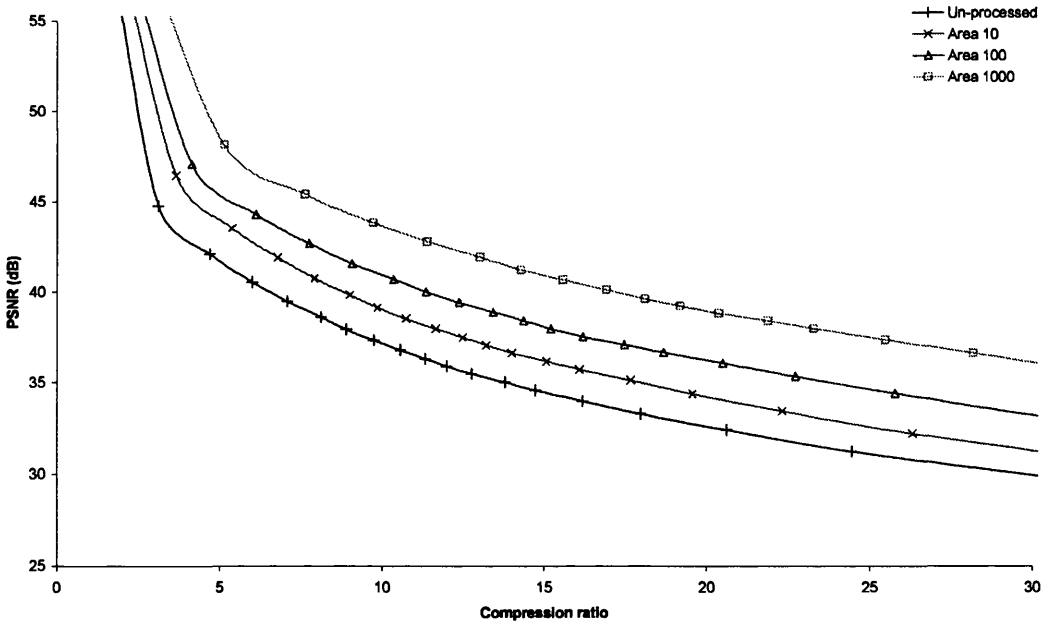


b) 8nn AF

Figure 13.39: JPEG rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the AF filter structure..

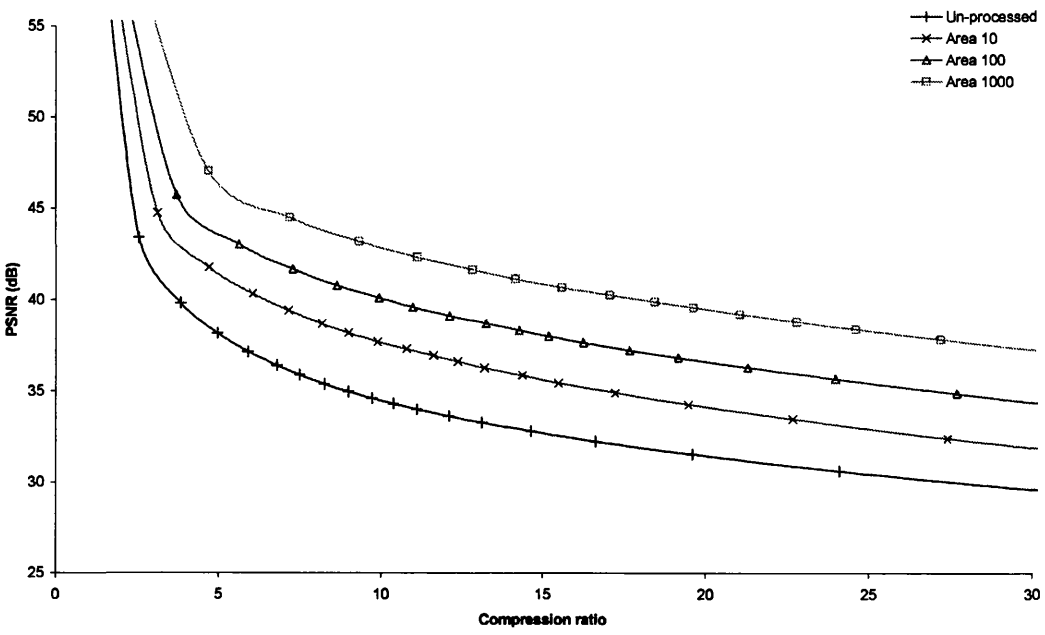


a) 4nn ASF

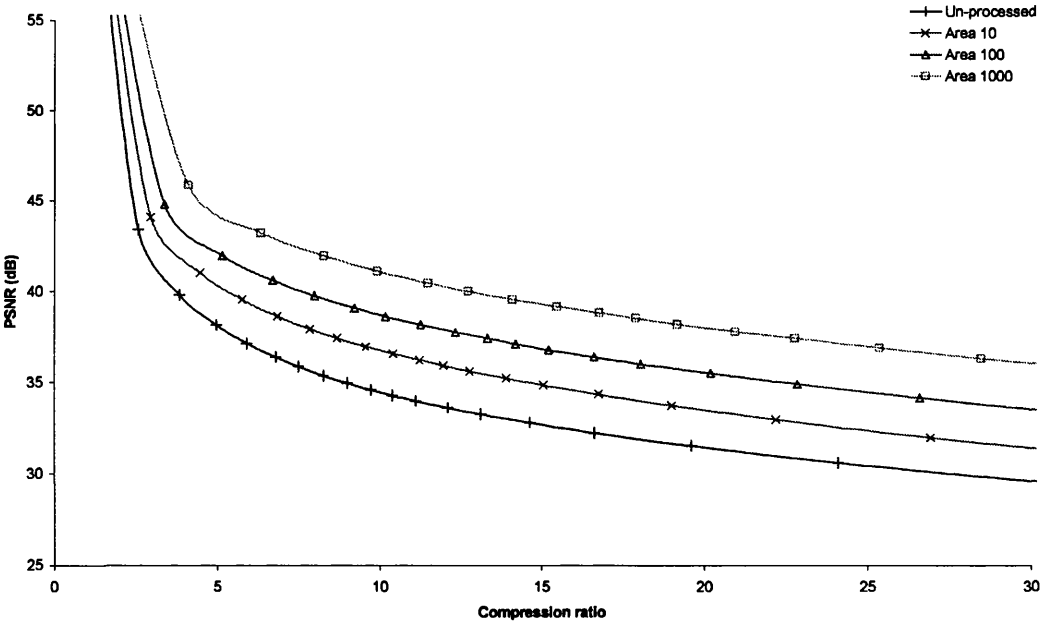


b) 8nn ASF

Figure 13.40: JPEG rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the ASF filter structure.

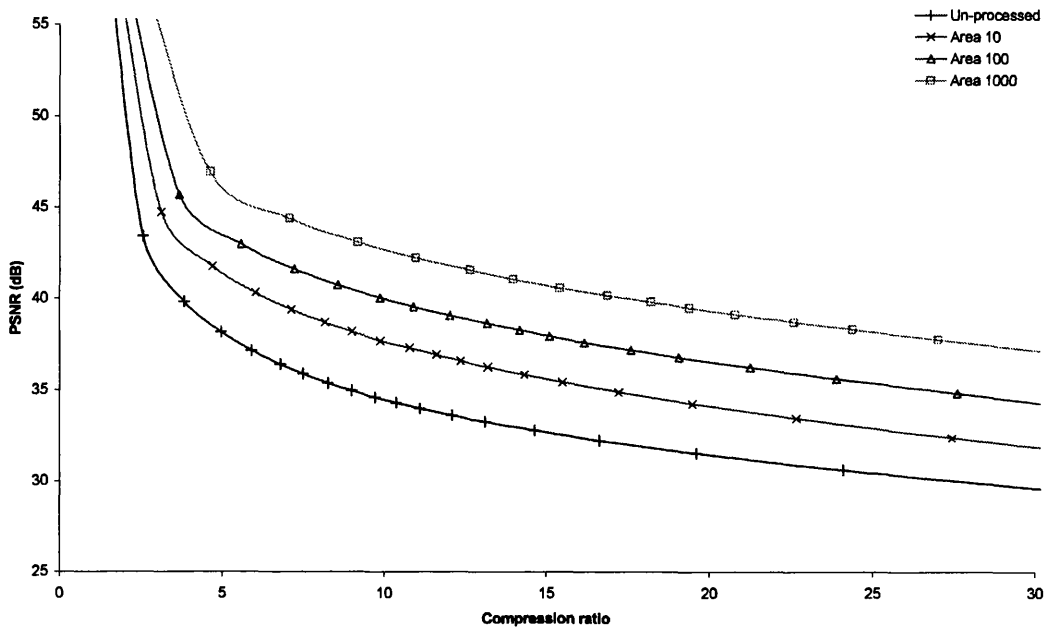


a) 4nn AF

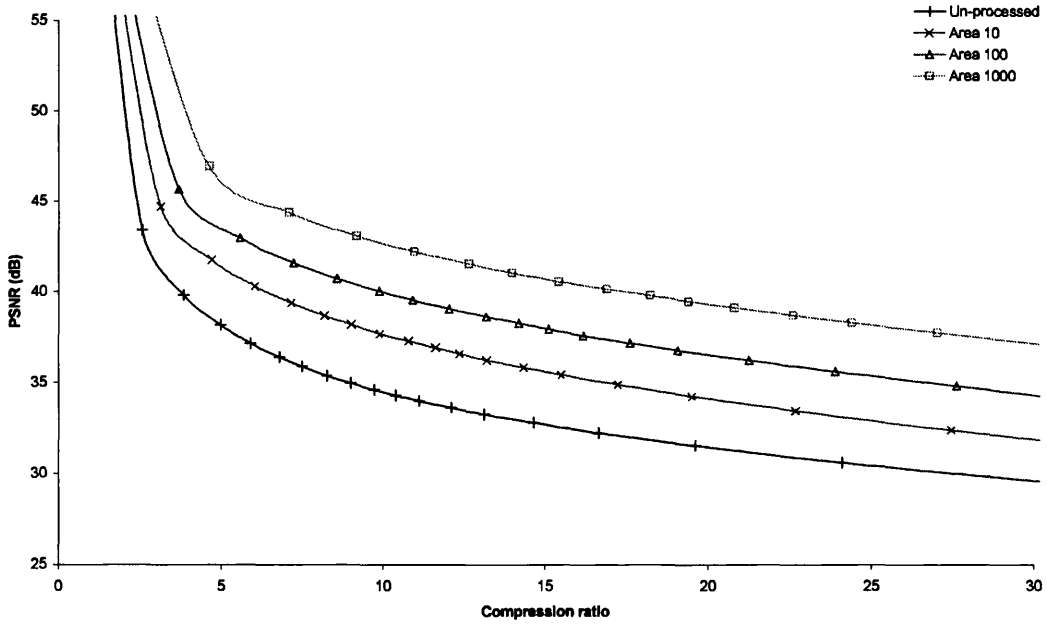


b) 8nn AF

Figure 13.41: JPEG rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the AF filter structure.

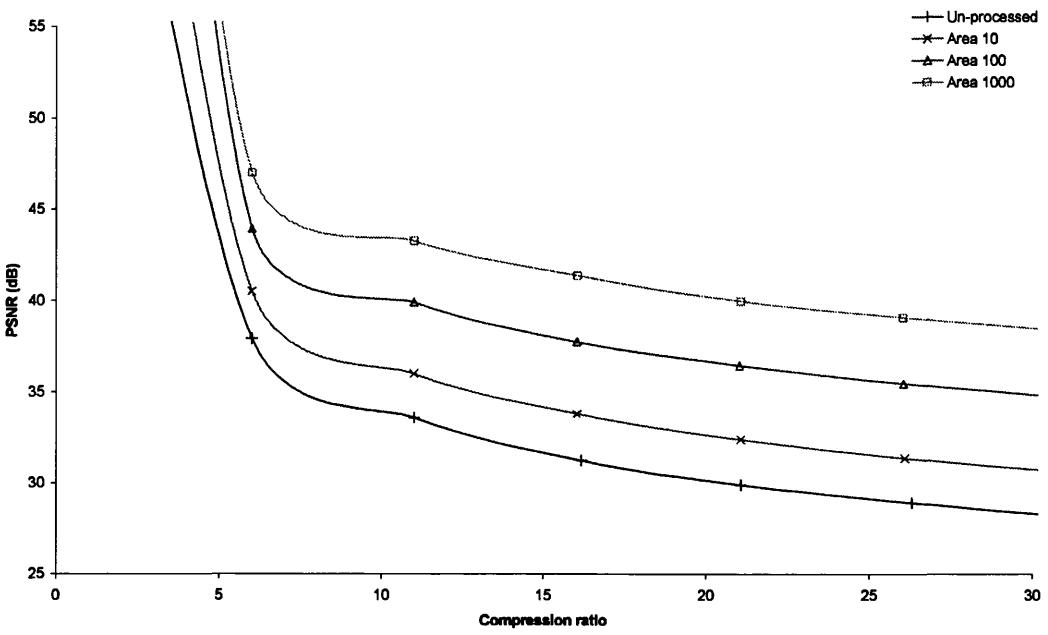


a) 4nn ASF

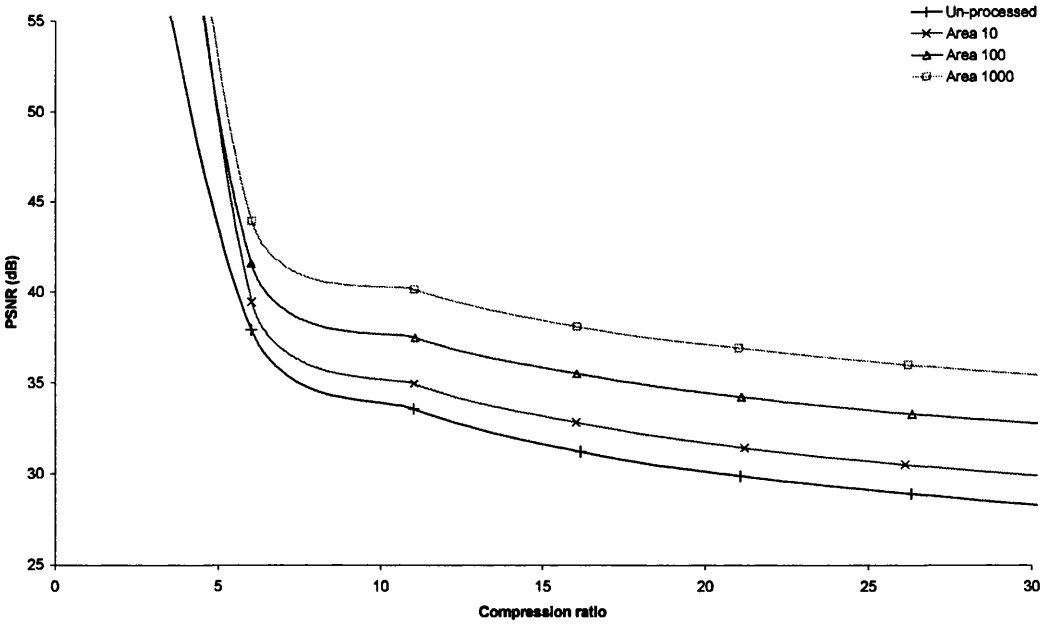


b) 8nn ASF

Figure 13.42: JPEG rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the ASF filter structure.

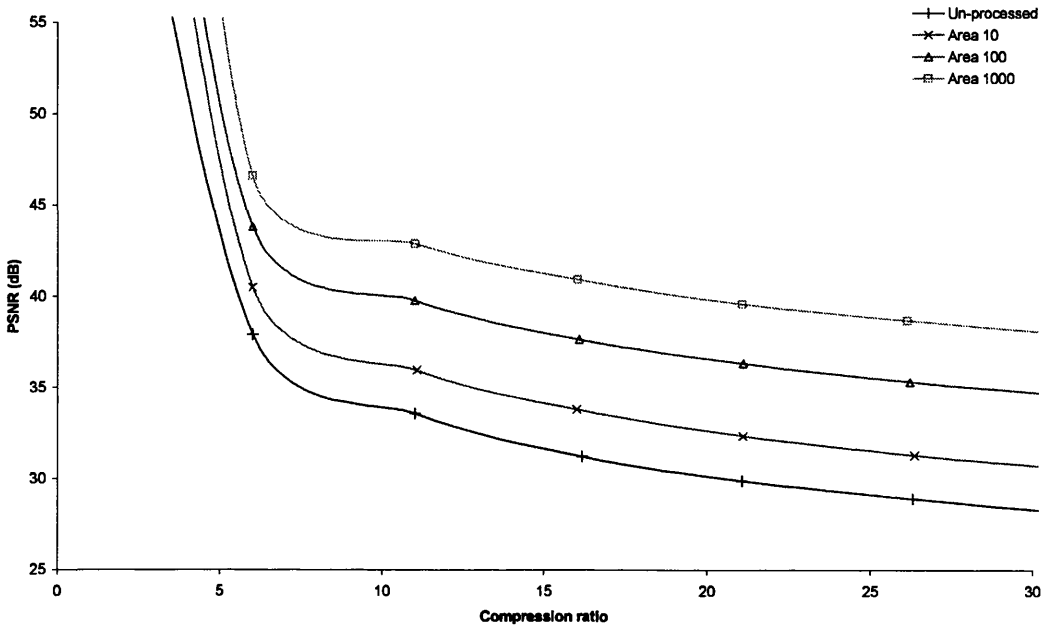


a) 4nn AF

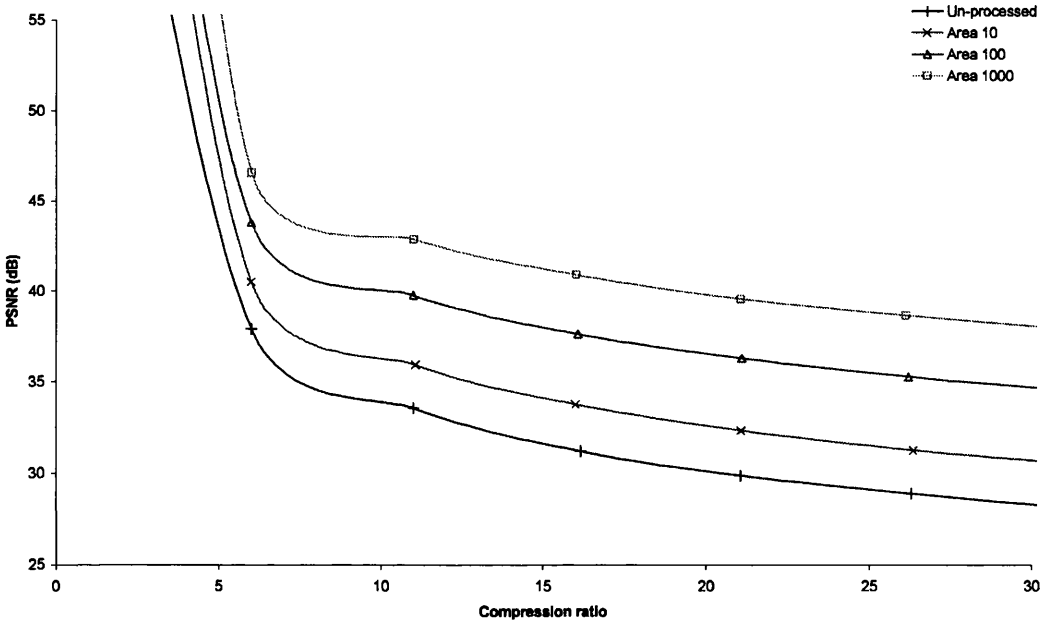


b) 8nn AF

Figure 13.43: JPEG 2000 rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the AF filter structure.

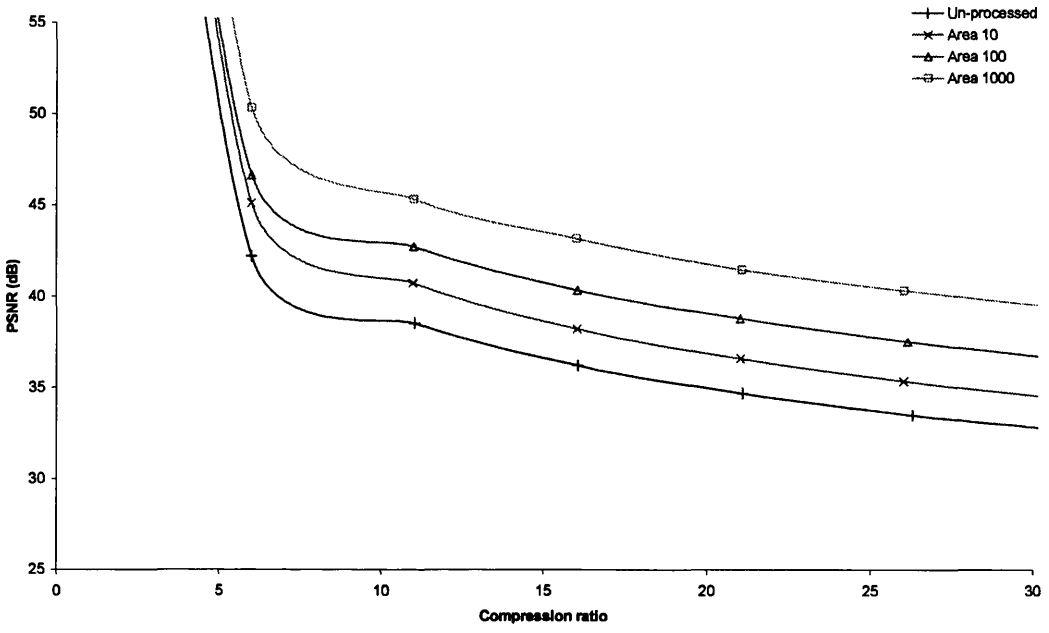


a) 4nn ASF

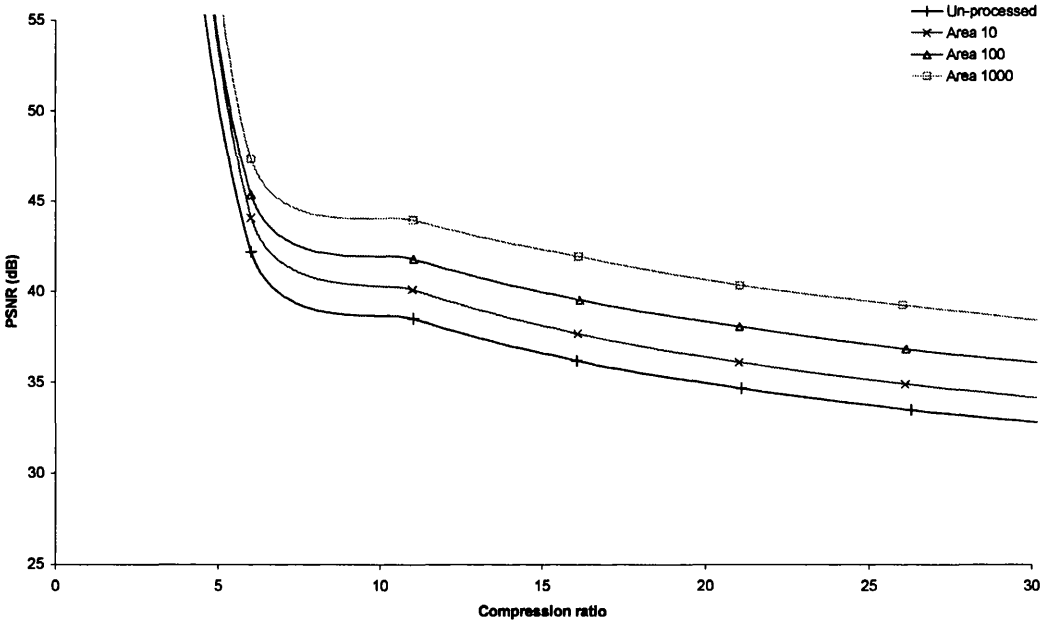


b) 8nn ASF

Figure 13.44: JPEG 2000 rate distortion graphs for the Barbara 2 image (8bit greyscale, 720 x 576) using the ASF filter structure.

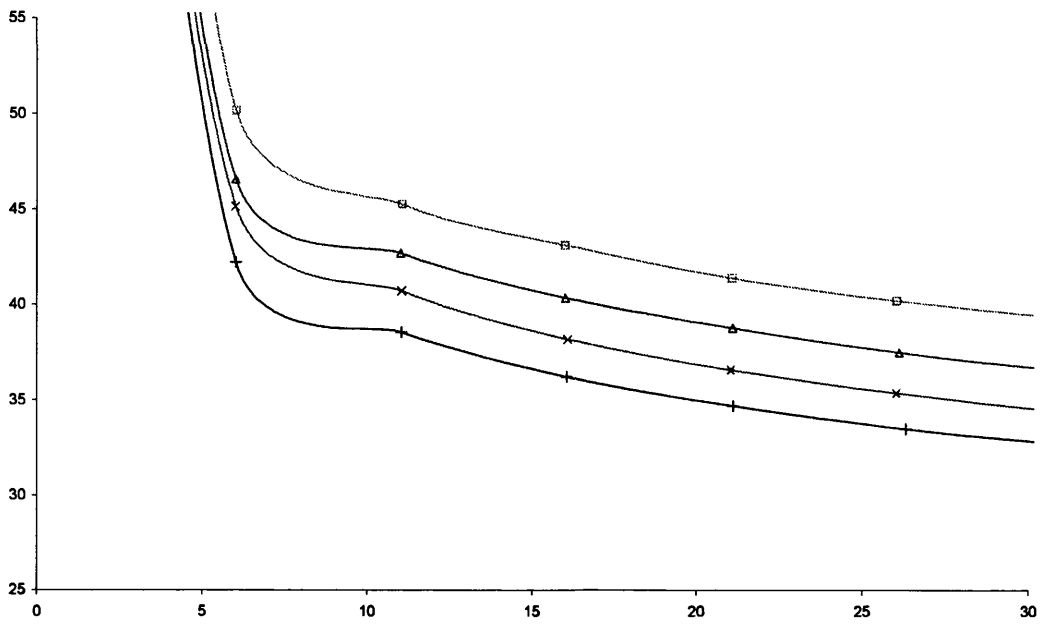


a) 4nn AF

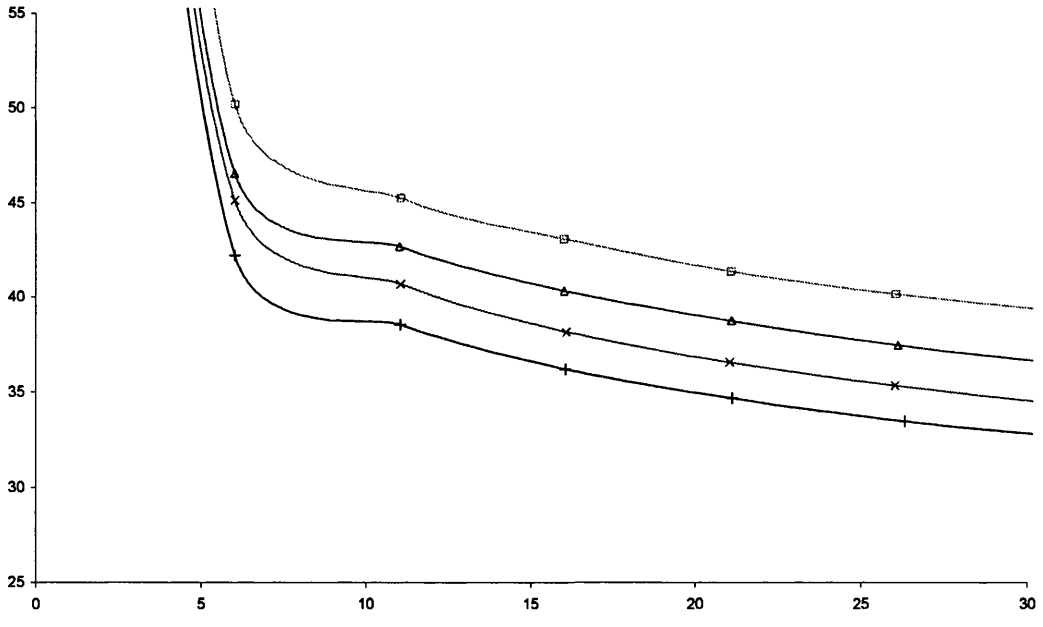


b) 8nn AF

Figure 13.45: JPEG 2000 rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the AF filter structure.

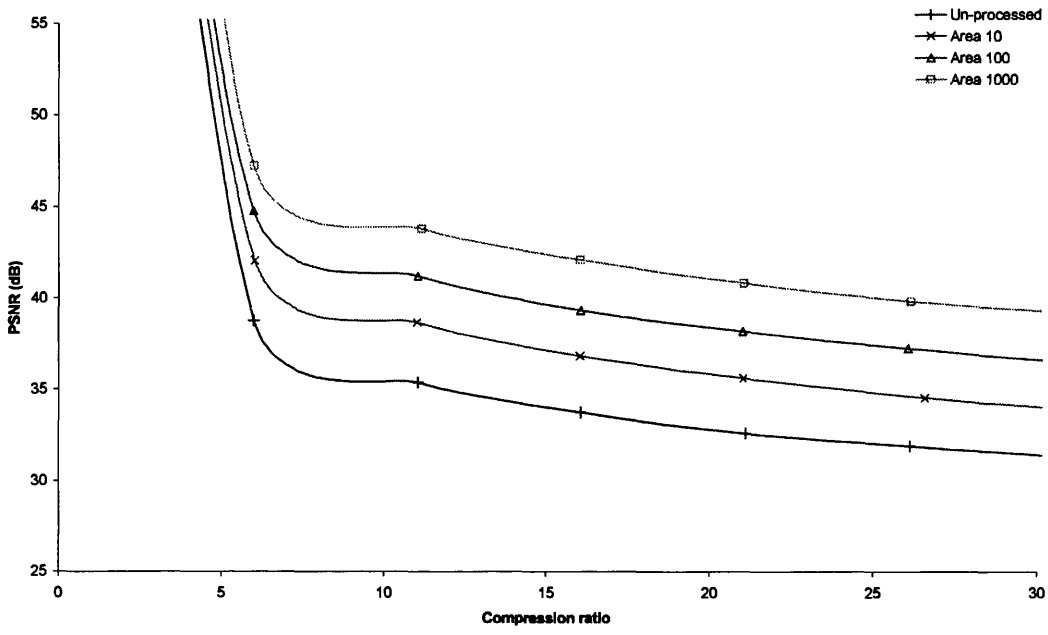


a) 4nn ASF

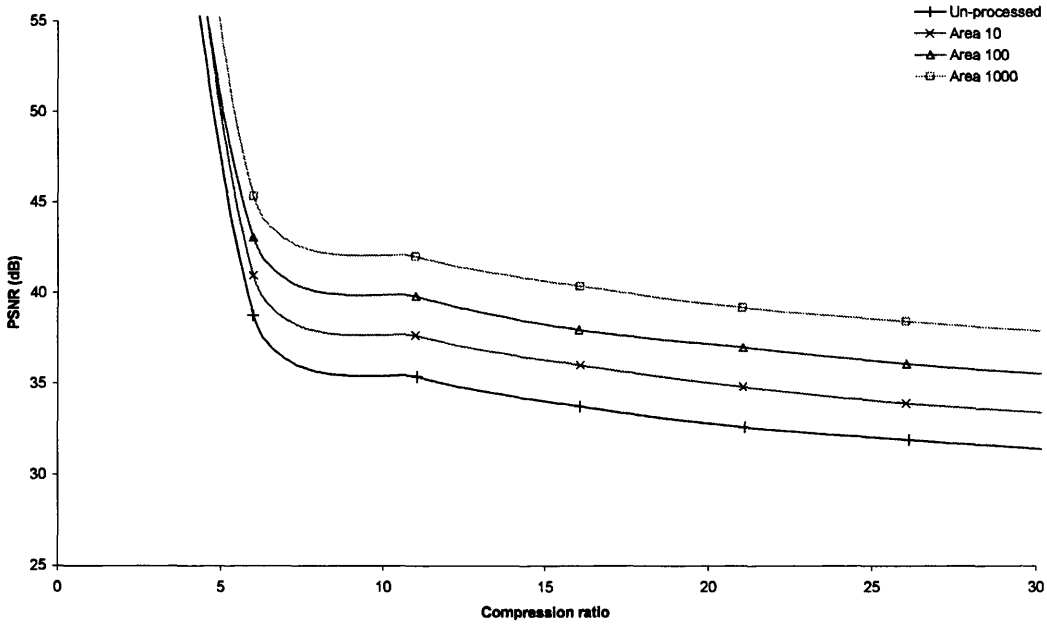


b) 8nn ASF

Figure 13.46: JPEG 2000 rate distortion graphs for the Boats image (8bit greyscale, 720 x 576) using the ASF filter structure.

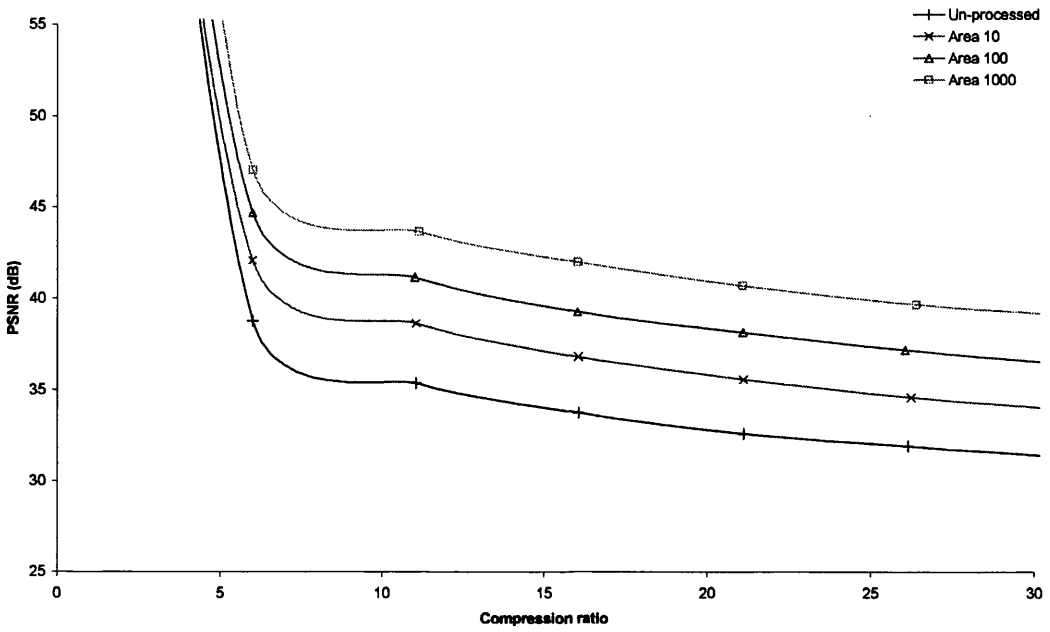


a) 4nn AF

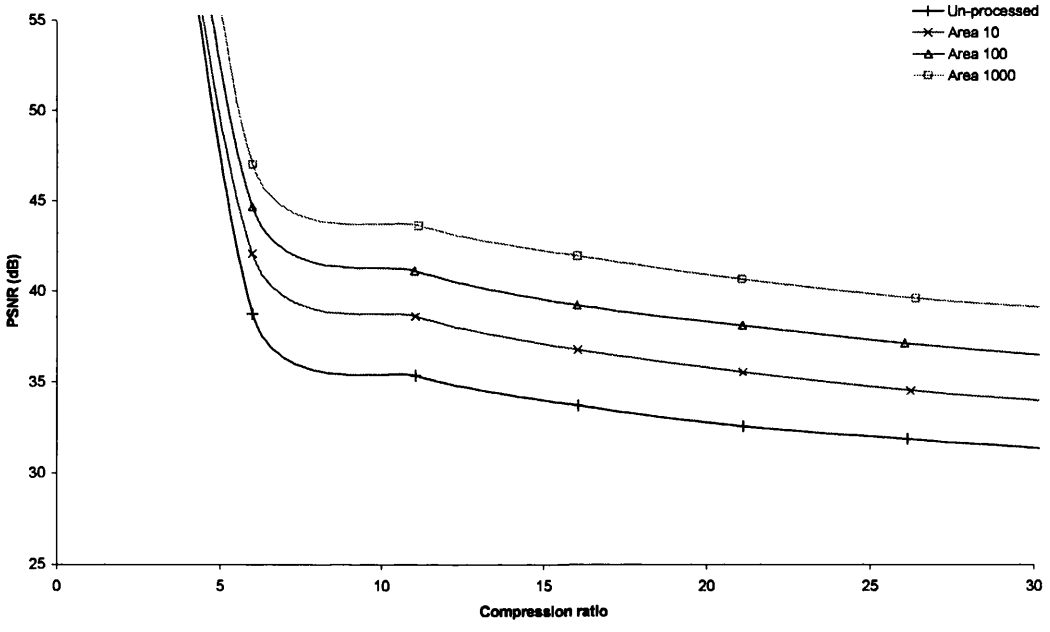


b) 8nn AF

Figure 13.47: JPEG 2000 rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the AF filter structure.



a) 4nn ASF



b) 8nn ASF

Figure 13.48: JPEG 2000 rate distortion graphs for the Goldhill image (8bit greyscale, 720 x 576) using the ASF filter structure.

14 Author's Publications

[1] N.Young and A.N. Evans, "Psycho-visually tuned area-morphology tools for improved image compression", *in Proceedings International Symposium on Mathematical Morphology (ISMM)*, Manly, Sydney, Australia, April 2002, pp. 185-195

[2] N.Young and A.N.Evans, "Image Noise Reduction using Attribute Morphology Filters", *in Proceedings IEEE EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP)* , Grado, Italy, June 2003.

[3] N.Young and A.N.Evans, "Digital video preprocessing with multi-dimensional attribute morphology, *in Proceedings IEE Visual Information Engineering (VIE) Conference*, University of Surrey, Guildford, Surrey, UK, July 2003, pp. 89 - 92

[4] N.Young and A.N.Evans, "Spatio-temporal attribute morphology filters for noise reduction in image sequences", in *Proceedings International Conference on Image Processing (ICIP)*, Barcelona, September 2003.

[5] N.Young and A.N.Evans, "Psychovisually tuned attribute operators for pre-processing digital video", accepted for publication in *Journal of Video and Image Signal Processing (VISP).*

15 References

[1] A. Emmerson, "Old Television", (Shire Publications Ltd, ISBN: 0-74780367-6, 1998)

[2] D.F.McLean, "Restoring Baird's image", *IEE Review*, September 2000, pp. 9-14.

[3] J.L.Baird, "Television, or Seeing by wireless", *Discovery*, 6, pp. 142-143., 1925.

[4] R.W.Burns, "British Television: The formative years", (Peter Peregrinus Ltd, ISBN: 0-86341-079-0, 1986)

[5] A.A.Campbell Swinton, "Distant Electric Vision", *Nature*, June 18th 1908, page 151.

[6] J.Slater, "Modern Television Systems to HDTV and beyond", (Pitman, ISBN: 0-273-03122-8, 1991)

[7] S.Ono, N.Ohta and T.Aoyama, "Super-High-Definition Images, Beyond HDTV", (Artec House, ISBN: 0-89006-674-4, 1995)

[8] R.H.Stafford, "Digital Television; Bandwidth Reduction and Communication Aspects" (John Wiley and Sons, ISBN: 0-471-07857-3, 1980)

[9] J.Watkinson, "The art of digital video", (Focal press, ISBN: 0-240-513-69-X, 1994)

[10] J.Watkinson, "Compression in video and audio", (Focal Press, 1995, ISBN: 0-240-51394-0)

[11] J.Watkinson, "An introduction to digital video", (Focal Press, ISBN: 0-240-51637-0, 2nd Edition 2001)

[12] J.G.Proakis, "Digital Communication", (McGraw-Hill, ISBN:0-07-113814-5, 1995)

[13] ITU-R Reccomendation BT.601-5, "Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios", International Telecommunications Union Recommendation, 1995

[14] "MPEG – Digital Television for all", (NTL, ISBN: 1-872567-07-X, 1995)

[15] F.W.Mounts "Frame-to-frame digital processing of TV pictures to remove redundancy" *in Picture Bandwidth compression*, Edited by T.S.Huang and O.J.Tretiak, 1972, pp. 653-672.

[16] F.Rocca, "Television bandwidth compression utilizing frame-to-frame correlation and movement compensation" in *Picture Bandwidth compression*, Edited by T.S.Huang and O.J.Tretiak, 1972, pp. 673-693.

[17] James.T.McIlwain, "An introduction to the biology of vision", (Cambridge University Press, ISBN: 0-521-49890-2, 1997)

[18] J.L.Mitchell, W.B.Pennebaker, C.E.Fogg & D.J.Leall, "MPEG Video Compression Standard", (Chapman and Hall, ISBN: 0-412-08771-5, 1996)

[19] G.Strang, T.Nyugen, "Wavelets and filterbanks", (Wellesley Cambridge Press, ISBN: 0-9614088-7-1, 1996)

[20] J.M.Shapiro, "Embedded Image Coding using Zerotrees of wavelet coefficients", *IEEE Transactions on Image Processing*, December 1993, **41**(12), pp. 3445-3462

[21] G.C.K.Abhayaratne and D.M.Monro, "Embedded to lossless image coding (ELIC)", *In Proceedings of Nordic Signal Processing Symposium (NORSIG 2000)*, Kolmarden, Sweeden, 13th-15th June 2000, pp. 255-258.

[22] G.C.K.Abhayaratne, "Lossless and nearly lossless digital video coding", PhD Thesis, Signal and Image Processing Group (SIPG), Department of Electronic Engineering, University of Bath, 2002.

[23] S.So and K.K.Paliwal, "Comparison of Quantisation Techniques for DCT-based Image Coding", *In Proceedings of Microelectronic Engineering Research Conference (MERC'01)*, 2001, Griffith University, Brisbane

[24] C.Amerijckx, M.Verleysen, P.Theissen and J.Legat, "Image Compression by Self-Organized Kohonen Map", *IEEE Transactions on Neural Networks*, May 1998, **9**(3), pp. 503-507.

[25] S.Heath, "Multimedia and Communications Technology", (Focal Press, ISBN: 0-240-51529-3, 1999)

[26] R.de Bruin and J.Smits, "Digital Video Broadcasting: Technology, Standards, and Regulations", (Artech House, ISBN: 0-89006-743-0, 1999)

[27] V.Bottreau, M.Benetiere, B.Feltz and B.Pesquet-Popescu, "A fully scalable 3D subband video codec", *International Conference on Image Processing (ICIP'01)*, October 7th-10th 2001, Thessaloniki, Greece, Volume 2, pp. 1017-1020.

[28] B.A.Wandell, "Foundations of Vision", (Sinauer Associates Inc, ISBN: 0-87893-853-2,1995)

[29] J.Emmett, "Perceptual Video Measurements", *Image Technology*, April 2001, pp. 8-12

[30] Martin.J.Tovée, "An introduction to the visual system", (Cambridge University Press, ISBN: 0-521-48339-5, 1996)

[31] C. Wang, S. Lee and L. Chang, "Designing JPEG quantization tables based on human visual system", *Signal Processing: Image Communication*, January 2001, **16**(5), pp. 501-506.

[32] J.F.Delaigle, C.Devleeschouwer, B.Macq and L.Langendijk, "Human Visual System Features Enabling Watermarking", *IEEE International Conference on Multimedia and Expo (ICME)*, 26th-29th August 2002, Lausanne, Switzerland

[33] P.Bao & D.Xu, "Complex wavelet based image mosaics using edge-preserving visual perception modelling", *Computers & Graphics*, 1999, **23**(3), pp. 309-321.

[34] B.A.Wandell, A.Gammal and B.Girod, "Common Principles of Image Acquisition: Systems and Biological Vision", *In Proceedings of the IEEE*, January 2002, **90**(1), pp. 5-17.

[35] R.C.Brainard "Subjective Effects of noise in television", in *Picture Bandwidth compression*, Edited by T.S.Huang and O.J.Tretiak, , pp. 31-46, 1972.

[36] ITU-R Recommendation BT.500-10, "Methodology for the subjective assessment of the quality of television pictures", International Telecommunications Union, 1995, March 2000.

[37] ITU-R Recommendation BT.1129-2, "Subjective assessment of standard definition digital television systems", International Telecommunications Union, February 1998.

[38] John Allnatt, "Transmitted-picture assessment", (John Wiley & Sons, ISBN: 0-471-90113-X, 1983)

[39] R.A.Rensink, "Seeing, sensing, and scrutinizing", *Vision research*, 2000, 40, pp. 1469-1487.

[40] A.B.Watson and L.Kreslake, "Measurement of visual impairment scales for digital video", *Proceedings of Human Vision and Electronic Imaging VI, SPIE*, 2001, Volume 4299, pp. 79-89

[41] J.Z.Levinson "Psychophysics and TV" in *Picture Bandwidth compression*, Edited by T.S.Huang and O.J.Tretiak, pp. 11-29, 1972.

[42] J.Serra, "The Birth of Mathematical Morphology", *In International Symposium on Mathematical Morphology (ISMM'02)*, 3rd-5th April 2002, Manly, Sydney, Australia, pp. 1-16.

[43] G.Matheron, "Random Sets and Integral Geometry", (John Wiley & Sons, ISBN: 0-471-57621-2, 1975)

[44] J.Serra, "Image Analysis and Mathematical Morphology", (Academic Press, ISBN: 0-12-637240-3, 1990)

[45] L.Floreby, F.Sattar, G.Salomonsson, "Image Enhancement by morphological pyramid decomposition and modified reconstruction", *In Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97)*, 21st-24th April 1997, Munich, Germany, Volume 4, pp. 2585-2588.

[46] N.R.Harvey and S.Marshall, "Film restoration using soft morphological filters", *In Proceedings 6th International Conference on Image Processing and its Applications (IPIA'97)*, 14th-17th July 1997, Trinity College , Dublin, Ireland, pp. 279-282

[47] R.C.Gonzalez & R.E.Woods, "Digital Image Processing", (Addison Wesley, ISBN: 0201180758, 2001)

[48] M.Sonka, V.Hlavac and R.Boyle, "Image Processing, Analysis and Machine Vision", (International Thomson Computer Press, ISBN:0-412-45570-6, 1993)

[49] J.A.Bangham & S.Marshall, "Image and signal processing with mathematical morphology", *Electronics & Communication Engineering Journal*, June 1998, **10**(3), pp117-128.

[50] N.R.Harvey and S.Marshall, "Restoration of archive film material using multi-dimensional soft morphological filters", *In Proceedings of the IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP'99)*, 20th-23rd June 1999, Antalya, Turkey, pp. 811-815.

[51] D.Wang & C.Labit, "Morphological spatio-temporal simplification for video image segmentation", *Signal Processing: Image Communication*, December 1997, **11**(2), pp. 161-170.

[52] C.Sun and S.Wu, "Automatic Segmentation of Fiducial Marks Using Attribute-based Mathematical Morphology", *Journal of Electronic Imaging*, April 2001, **10**(2), pp. 560-566.

[53] H.Gao, W.Siu, C.Hou, "Improved Techniques for Automatic Image Segmentation", *IEEE Transactions on circuits and systems for video technology*, December 2001, **11**(12), pp. 1273-1280.

[54] D.Schonfeld and J.Goutsias, "Optimal Morphological Pattern Restoration from Noisy Binary Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, January 1991, **13**(1), pp. 14-29.

[55] S.Saryazdi, V.Haese-Coat and J.Ronsin, "Image representation by a new optimal non-uniform morphological sampling", *Pattern Recognition*, June 2000, **33**(6), pp. 961-977.

[56] F.Cheng and A.N.Venetsanopoulos, "Fast, Adaptive Morphological Decomposition for image compression", *In Proceedings of the 25th Annual conference on information sciences and systems*, 20th-22nd March 1991, Baltimore, MD, pp. 35-40.

[57] K. Sivakumar, J.Goustasis, "Morphological size densities: application to optimal texture classification and noise filtering", *IEEE Workshop on Nonlinear Signal and Image Processing*, 1997

[58] G.J.F.Banon and S.D.Faria, "Morphological Approach for Template Matching", *In Proceedings of Brazilian Symposium of Graphical Computation and Picture processing (SIBGRAPI'97)*, 14th-17th October 1997, Fields of the Jordão, SP

[59] J.G.Verly and R.L.Delanoy, "Adaptive Mathematical Morphology for Range Imagery", *IEEE Transactions on Image Processing*, April 1993, **2**(2), pp. 272-275.

[60] B.S.Daya Sagar and K.S.R.Murthy, "Generation of a fractal landscape using nonlinear mathematical morphological transforms",*Fractals*, 2000, **8**(3), pp. 267-272.

[61] P.Soille, "Morphological Image Analysis: Principles and Applications", (Springer, ISBN: 3-540-65671-5, 1999)

[62] H.J.A.M.Heijmans, "The Algebraic Basis of Mathematical Morphology: Part I. Dilations and Erosions", *Computer Vision, Graphics, and Image Processing (CVGIP)*, June 1990, **50**(3), pp. 245-295.

[63] C.Ronse and H.J.A.M.Heijmans, "The Algebraic Basis of Mathematical Morphology: Part II. Openings and Closings", *Computer Vision, Graphics, and image Processing*, 1991, **54**(1), pp. 74-97.

[64] S.T.Acton, D.O.Mukherjee, "Scale space classification using area morphology", *IEEE Transactions on image Processing*, April 2000, **9**(4), pp. 623-635.

[65] R.Hutt, "Grey-level Morphology Based Segmentation of MRI of the Human Cortex and Applications on Visualisation", PhD Thesis, Centre for Image Analysis, Uppsala University, 2001, ISBN 91-576-5654-1, ISSN 1404-0301.

[66] R.J.Lapeer, A.C.Tan and R.Aldridge, "A combined approach to 3D medical image segmentation using marker-based watersheds and active contours: the active watershed method", *Medical Image understanding and analysis*, 22nd-23rd July 2002, University of Portsmouth

[67] B. Dogdas, D.Shattuck and R.M.Leahy, "Segmentation of the Skull in 3D Human MR Images Using Mathematical Morphology", *In Proceedings of SPIE Medical Imaging Conference (MIC)*, 10th-16th November 2002, Norfolk, Virginia, Volume 4684, pp. 1553-1562.

[68] K.Park and C.Lee, "Scale Space using mathematical morphology", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, November 1996, **18**(11), pp. 1121-1126.

[69] J.A.Bangham, P.Chardaire, C.J.Pye and P.D.Ling , "Multiscale nonlinear decomposition: the sieve decompostion theorem", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, May 1996, **18**(5), pp. 529-539.

[70] J.A.Bangham, P.D.Ling and R.Harvey, "Scale-Space from nonlinear filters", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1996, **18**(5), pp. 520-528.

[71] A.Cunha, R.Teixeira, L.Velho, "Discrete Scale Spaces", *International Symposium on Mathematical Morphology (ISMM)*, 3rd-5th April 2002, Manly, Sydney, Australia, pp. 241-251.

[72] H.J.A.M.Heijmans, "Morphological Scale-Spaces, Scale-Invariance, And Lie Groups", *International Symposium on Mathematical Morphology (ISMM)*, 3rd-5th April 2002, Manly Sydney, Australia, pp. 253-263.

[73] J.Crespo, R.Schafer, F.Meyer, J.Serra & C.Gretin, "The flat zone approach: A general low-level region merging segmentation method", *Signal processing*, 1997, **62**(1), pp. 37-60.

[74] S.R.Sternberg, "Grayscale Morphology", *Computer vision, graphics and image processing*, 1986, 35, pp. 333-355.

[75] A.Meijster and M.H.F.Wilkinson, "Fast computation of morphological area pattern spectra", *In Proceedings of International Conference on Image Processing (ICIP'01)*, 7th-10th October 2001, Thessaloniki, Greece, pp 668-671.

[76] F.Cheng and A.N.Venetsanopoulos, "An Adaptive Morphological Filter for Image Processing", *IEEE Transactions on Image Processing*, October 1992, **1**(4), pp. 533-539.

[77] L.Vincent, "Morphological grayscale reconstruction in image analysis: applications and efficient algorithms", *IEEE Transactions on image processing*, April 1993, 2(2), pp176-201.

[78] L.Vincent, "Grayscale area openings and closings, their efficient implementation and applications", *EURASIP Workshop on Mathematical Morphology and its Applications to Signal Processing*, May 1993, Barcelona, Spain, pp. 22-27.

[79] E.J.Breen and R.Jones, "Attribute Openings, Thinings, and Granulometries", *Computer Vision and Image Understanding*, November 1996, **64**(3), pp. 377-389.

[80] P.Salembier, F.Meyer, P.Brigger and L.Bouchard, "Morphological operators for very low bit rate video coding", *In Proceedings of IEEE International Conference on Image Processing*, September 1996, Lausanne, Volume III, pp. 659-662.

[81] N.Young and A.N. Evans, "Psycho-visually tuned area-morphology tools for improved image compression", *in Proc. International Symposium on Mathematical Morphology (ISMM)*, 3rd-5th April 2002, Manly, Sydney, Australia, pp. 185-195.

[82] N.Young and A.N.Evans, "Image Noise Reduction using Attribute Morphology Filters", *in Proceedings IEEE EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP)* , Grado, Italy, June 2003.

[83] N.Young and A.N.Evans, "Digital video preprocessing with multi-dimensional attribute morphology, *in Proceedings IEE Visual Information Engineering (VIE) Conference*, University of Surrey, Guildford, Surrey, UK, July 2003, pp. 89 - 92

[84] M.L.Comer and E.J.Delp, Morphological operations for color image processing, *Journal of Electronic Imaging*, July 1999, **8**(3), pp 279-289.

[85] R.A.Peters, "Mathematical morphology for angle-valued images", *In Proceedings of International conference on Electronic Imaging*, Society of Photo-optical Instrumentation, February 1997, San Jose, CA

[86] M.Saenz, R.Oktem, K.Egiazarian and E.J.Delp, "Color image wavelet compression using vector morphology", In Proceedings of the European Signal Processing Conference (EUSIPCO), 5th-8th September 2000, Tampere, Finland

[87] V.Caselles, G.Sapiro and D.H.Chung, "Vector Median Filters, Morphology, and PDE's: Theoretical connections", *In Proceedings International Conference on Image Processing (ICIP'99)*, 1999, Volume 4, pp. 177-181.

[88] S.W.Smith, "The Scientist and Engineer"s Guide to Digital Signal Processing", (California Technical Publishing, 1999, ISBN: 0-9660176-7-6)

[89] J.B.O'Neal, "Noise Considerations in bit rate compression" in *Picture Bandwidth compression*, Edited by T.S.Huang and O.J.Tretiak, pp. 197-214, 1972.

[90] J.D.Martin, "Signals & Processes: A Foundation Course", (Pitman, ISBN: 0-273-03256-9, 1991)

[91] S.I.Olsen, "Estimation of Noise in Images: An Evaluation", *Computer Vision, Graphics, and Image Processing (CVGIP): Graphical Models and Image Processing*, July 1993, **55**(4), pp. 319-323

[92] P. Meer, J. Jolion and A. Rosenfeld, "A Fast Parallel Algorithm for Blind Estimation of Noise Variance", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, February 1990, **12**(2), pp. 216-222.

[93] A. Amer, A. Mitiche and E. Dubois, "Reliable and fast structure-oriented video noise estimation",

[94] A.V.Oppenheim, R.W.Schafer and T.G.Stockham, "Nonlinear filtering of Multiplied and Convolved Signals", *In Proceedings of the IEEE*, August 1968, **56**(8), pp. 1264-1291.

[95] B.I.Justusson, "Median Filtering: Statistical Properties", *In Topics in Applied Physics: Two-Dimmensional Digital Signal Processing II*, Edited by T.S.Huang, pp. 161-196, Springer-Verlag, ISBN: 3-540-10359-7,1981

[96] S.G.Tyan, "Median Filtering: Determistic Properties", *In Topics in Applied Physics: Two-Dimmensional Digital Signal Processing II*, Edited by T.S.Huang, pp167-217, Springer-Verlag, ISBN: 3-540-10359-7,1981

[97] K.J.Willner, P.Kuosmanen, V.V.Lukin and A.B.Pogrebniak, "Nonlinear filters and rapidly increasing/decreasing signals corrupted with noise", *IEEE Workshop on Nonlinear Signal and Image Processing (NSIP'97)*, 8th-10th September 1997, Mackinac Island, Michigan, USA.

[98] A.B.Hamza and H.Krim, "Nonlinear image filtering: trade-off between optimality and praticality", *In Proceedings of International Conference on Image Processing (ICIP'01)*, 2001, October 7th-10th 2001, Thessaloniki, Greece, pp. 126 – 129.

[99] J.T.Astola and T.G.Campbell, "On Computation of the Running Median", *IEEE transactions on acoustics, speech and signal processing*, April 1989, **37**(4), pp. 572-574.

[100] T.S.Huang, G.J.Yang and G.Y.Tang, "A Fast Two-Dimensional Median Filtering Algorithm", *IEEE transactions on acoustics, speech and signal processing*, February 1979, **27**(1), pp. 13-18.

[101] T.A.Nodes and N.C.Gallagher, "Median Filters: Some Modifications and Their Properties", *IEEE transactions on acoustics, speech and signal processing*, October 1982, **30**(5), pp. 739-746.

[102] X.Xu and E.L.Miller, "Adaptive Two-pass Median Filter to Remove Impulsive noise", *International Conference on Image Processing (ICIP'02)*, 14th – 17th September, 2002, Barcelona, Spain, pp 808-811.

[103] V.I.Ponomaryov, F.J.G.Funes and O.B.Pogrebnyak, "Novel fine detail preserving filter in the image processing", *In Proceedings of Nordic Signal Processing Symposium (NORSIG 2000)*, 13th-15th June 2000, Kolmarden, Sweeden, pp. 137 – 140.

[104] A.B.Hamza and H.Krim, "Image Denoising: A Nonlinear robust Statistical Approach", *IEEE Transactions on signal processing*, December 2001, **49**(12), pp. 3045-3054.

[105] L.Yin, R.Yang, & M.Gabbouj, "Weighted median filters: A tutorial", *IEEE Transactions on circuits and systems II: Anaolg and digital signal processing*. 1993, **43**(3), pp. 157-189.

[106] L.S.Davis and A.Rosenfeld, "Noise Cleaning by Iterated Local Averaging", *IEEE Transactions on systems, man, and cybernetics*, September 1978, **8**(9), pp. 705-710.

[107] N.Vasconcelos & F.Dufaux, "Pre and post filtering for low bit-rate video coding", *In Proceedings of IEEE International Conference on Image Processing (ICIP'97)*, 1997, Santa Barbara, USA, pp. 291-294.

[108] G.L.Bretthorst, "Lecture Notes in Statistics", (Springer-Verlag, ISBN: 0-387-96871-7. 1988)

[109] H.Norell, B.Oelmann and Y.Xu, "Pre and post filtering for low bit-rate video coding", *In Proceedings of Nordic Signal Processing Symposium (NORSIG 2000),* Kolmarden, Sweeden, 13th-15th June 2000, pp. 21-24.

[110] L.Yan, "Noise reduction for MPEG type of CODEC", *In Proceedings of International Conference on Accoustices, Speech and Signal Processing (ICASSP'94)*, April 1994, Adelaide, pp. 429-432.

[111] V. Zlokolica, W.Phillips and D. Van De Vill, "A New non-linear filter for video processing", *In Proceedings of IEEE Benelux Signal Processing Symposium (SPS 2002)*, 21st-22nd March 2002, Leuven, Belgium, pp. 221-224.

[112] M.Haritopoulos, H.Yin and N.M.Allinson, "Multiplicative noise removal using self-organizing maps", *3rd International Conference on Independent Component Analysis and Signal Seperation*, 9th-13th December 2001, San Diego, California, pp. 206 – 211.

[113] T.Law, H.Itoh & H.Seki, "Image filtering, edge detection, and edge tracing using fuzzy reasoning", *IEEE Transactions on pattern analysis and machine intelligence (PAMI)*, 1996, **18**(5), pp481-491.

[114] M.Munegasu, Y.Wada, & T.Hinamoto, "Edge preserving smoothing by adaptive nonlinear filters based on fuzzy control laws", *International Conference on Image Processing (ICIP'96),* 16th – 19th September 1996, Laysanne, Switzerland, pp. 785-788.

[115] G.Ramponi, "A rational edge-preserving smoother", *International Conference on Image Processing (ICIP'95),* 23rd – 26th October 1995, Washinton DC, USA, pp.151-15.

[116] E.Dubois and S.Sabri, "Noise Reduction in Image Sequences Using Motion-Compensated Temporal Filtering", *IEEE transactions on Communications*, July 1984, **32**(7), pp. 826-831.

[117] H.Tsuji, T.Sakatani, Y.Yashima and N.Kobayashi, "A nonlinear spatio-temporal diffusion and its application to prefiltering in MPEG-4 video coding", *International Conference on Image Processing (ICIP'02),* 14th – 17th September, 2002, Barcelona, Spain, pp. 85-88.

[118] P.Perona and J.Malik, "Scale-Space and Edge Detection Using Anisotropic Diffusion", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI),* July 1990, **12**(7), pp. 629-639.

[119] S.Tsekeridou, C.Kotropoulos and I.Pitas, "Morphological signal adaptive median filter for still image and image sequence filtering", *In Proc. of IEEE Int. Symposium on Circuits and Systems(ISCAS'98)*, 31st May-3rd June 1998, California, USA, Volume 4, pp. 21-24.

[120] R.A.Peters, "A new algorithm for image noise reduction using Mathematical Morphology", *IEEE Transactions on image Processing*, May 1995, 4(3), pp 554-568.

[121] A.N.Hirani and T.Totsuka, "Combining Frequency and Signal domain Information for Fast Interactive Image Noise Removal", *In Proceedings of SIGGRAPH'96*, New Orleans, LA, 4th-9th August 1996, pp. 269-276.

[122] P. A. Maragos & R. W. Schafer. "Morphological filters-part I: Their set-theoretic analysis and relations to linear shift-invariant filters", *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1987, **35**(8), pp. 153-1169.

[123] P. A. Maragos and R. W. Schafer. "Morphological filters-part II: Their relations to median, order statistic, and stack filters", *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1987, **35**(8), pp. 1170-1184.

[124] R.C.Leinecker and K.Smith, "Visual C++ 6.0 Bible", (Hungry Minds Inc, ISBN: 0764532286, 1998)

[125] W.H.Murray, C.H.Pappas, "Data Structures With STL", (Prentice Hall PTR, ISBN: 0-13-028927-2, 2001)

[126] H.Heijmans, "Mathematical Morphology: Basic Principles", In Proceedings of Summer School on Morphological Image and Signal Processing, Zakopane, 1995.

[127] J.Barrera and R.Hirata, "Fast Algorithms to compute the elementary operators of Mathematical Morphology", *Symposium on Computer Graphics and Image Processing (SIBGRAPI'97)*, 14th –17th October 1997, Fields of the Jordão, pp. 163-170

[128] L.Vincent, "Morphological Grayscale Reconstruction: Definition, Efficient Algorithm and Applications in Image Analysis", *In Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, June 1992, Champaign IL, pp. 633-635.

[129] SDC Morphology Toolbox, www.mmorph.com

[130] P. Davies, "The Indispensable guide to C", (Addison-Wesley, ISBN: 0-201-62438-9, 1996)

[131] T. Niemann, "Sorting and Searching Algorithms", ePaper Press, www.ePaperPress.com

[132] A.Meijster and M.H.F.Wilkinson, "A Comparison of Algorithms for Connected Set Openings and Closings", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, April 2002, 24(4), pp. 484-494.

[133] S.T.Acton, Fast Algorithms for Area Morphology, Digital Signal processing, 2001, 11, pp. 187-203.

[134] P.Salembier, A.Oliveras and L.Garrido, "Anti-extensive Connected Operators for Image and Sequence Processing", *IEEE Transactions on Image Processing*, April 1998, 7(4), pp. 555-570.

[135] G. Sankar, "Efficient Implementation of Hierarchical Queue for Morphological Image Processing",

[136] E.R.Urbach & M.H.F.Wilkinson, "Shape only granulometries and gray-scale shape filters", *In Proc. International Symposium on Mathematical Morphology (ISMM)*, 3rd-5th April 2002, Manly, Sydney, Australia, pp. 305-314.

[137] R.E.Tarjan, "Efficiency of a Good But Not Linear Set Union Algorithm", *Journal of the Association for Computing Machinery (ACM)*, April 1975, **22**(2), pp. 215-225.

[138] M.H.F.Wilkinson and J.B.T.M.Roderdink, "Fast Morphological attribute operations using Tarjan's union-find algorithm", *In Proceedings of the International Symposium on Mathematical Morphology (ISMM)*, June 2000, Palo Alto, pp 311-320.

[139] L.Vincent, "Area Opening and Closing for Grayscale Images", *In Proceedings NATO Shape in Picture Workshop*, September 1992, Driebergen, The Netherlands, Springer-Verlag, pp. 197-208.

[140] M.Grimaud, "A new measure of contrast: the dynamics", *Proceedings of Image Algebra and Morphological Image Processing, SPIE*, July 1992, San Diego, California, USA, Volume 1769, pp. 292-305, 1992.

[141] C.Vachier and L.Vincent, "Valuation of Image Extrema using Alternating Filters by Reconstruction", In Proceedings Image Algebra and Morphological Processing, SPIE, July 1995, San Diego California, Volume 2568, pp. 94-103.

[142] M.Bowers, "Scalable Wavelet Coding", PhD Thesis, Signal and Image Processing Group (SIPG), Department of Electronic Engineering, University of Bath, 2001.

[143] P.Meer, J.Jolion and A.Rosenfeld, "A Fast Parallel Algorithm for Blind Estimation of Noise Variance", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, February 1990, **12**(2), pp. 216-223.

[144] P.Salembier & J.Serra, "Flat zones, connected operators and filters by reconstruction", *IEEE Transactions on image processing*, August 1995, **4**(8), pp. 1153-1159.

[145] A.Croft, R.Davison & M.Hargreaves, "Engineering Mathematics", (Addison Wesley, ISBN: 0-201-17557-6, 1992)

[146] A.Croft, R.Davison and M.Hargreaves, "Introduction to Engineering Mathematics", (Addison Wesley, ISBN: 0-201-62442-7, 1995)